Smart School Improvement Online Survey System

by

Gang Zhang

Bachelor of Science, XinJiang University, 1983

A Project

Submitted to the Graduate Faculty

of the

University of North Dakota

in partial fulfillment of the requirements

for the degree of

Master of Science

Grand Forks, North Dakota

August 2003

This report, submitted by Gang Zhang in partial fulfillment of the requirements for the Degree of Master of Science from the University of North Dakota, has been read by the Faculty Advisor under whom the work has been done and is hereby approved.

Signature _____

Date _7/25/03_____

This project document meets the standards for appearance, conforms to the style and format requirements of the Computer Science Department of the University of North Dakota, and is hereby approved.

Signature _____

Date _7/28/03_____

PERMISSION

Title: Design and Implement Smart School Improvement Online Survey System

Department: Department of Computer Science

Degree: Master of Science

In presenting this report in partial fulfillment of the requirements for a graduate
degree from the University of North Dakota, I agree that Department of Computer
Science shall make it freely available for inspection. I further agree that permission for
extensive copying for scholarly purposes may be granted by the professor who supervised
my work or, in his absence, by the Chairperson of the Department.

Signature _____

Date _____07 / 25 / 2003_____

Table of Contents

List of Figures

List of Tables

Acknowledgement

Thanks to Dr. Wiggen, who is my academic advisor, gave me guide, advice, and help during my master program study in computer science, especially in this project.

I also would like to thank Dr. Simanton, Ms. Solseng, and all other staff of Bureau of Educational Services and Applied Research for their support this project and care, kindness to me.

To My Son, Ruo-Zhen and My Wife, Xin

Abstract

In this report, we describe the design and implementation of a computer application for Smart School Improvement Online Survey System. This computer application is named as a Smart Online Survey, because it can help schools to make better decisions about survey forms and the survey process based on their requirements and the experience that is embedded in this SSIOSS. Also it can lead a survey company working on a task, such as school improvement survey to the best process by using data analysis software, such as SPSS.

This project applied software engineering techniques and principals throughout the programming process. We designed and implemented this decision-making program by getting school requirement, based on the information of experience on statistic data analysis, and finally achieve the desired result. This makes the system more intelligent and flexible.

Chapter I: Introduction

1.1 Background

School accreditation and evaluation organizations such as NCA COS (North Central Association of Commission of Schools) consisting of 19 states in the north central region states requires schools to demonstrate school improvement with solid data. The major function and goal as stated in its mission statement is "NCA is a dynamic school accreditation and evaluation organization that protects the public trust and is proactive in promoting a system of education that enhances student learning; fosters healthy, creative and innovative human beings; prepare students to live and learn in an ever changing diverse world; and ensures successful schooling transitions for its learners through the provisions of standards and evaluation services for schools" (NCA Quarterly, 1998).

Each year NCA COS accredits schools by its standards in the area of School Improvement Plan (SIP), Information System (IS), Process of Schooling (PS), Vision, Leadership-Governance, and School Community (VL), Resources and Allocation (RA) (NCA Standard and Criteria for Elementary, Middle Level, Secondary, and Unit Schools, 2000). Whichever above endorsement is selected by a school to get the NCA COS accreditation, the goal of the NCA COS school improvement process in that school is to improve the achievement of the students. Improvement must be documented in such a way that the results clearly indicate if the methodology of the improvement plan is a good plan employed for pursuing the content of the selected goals. The old tools of education-intuition, teaching philosophy, and personal experience- do not seem to be enough anymore to show that the school is getting good results. Collecting valid data reflecting

student learning, effectiveness of teaching, administration and infrastructure improvement has become widespread. To ensure valid data requires approaches to collect reliable and valid information without too much interference with the normal school routine. One efficient method for administering such a task is to collect data on-line.

There are several problems in the process of school improvement data collection on line. First of all, survey companies need to know the proper items a particular school would like to be included in their data collection. Then they want to know how to ensure the whole on-line process to satisfy their requirements, including the groups of people involved in the survey, which items should be asked, how to range scope of survey data, which method to use to collect and transfer data from customers to the survey companies, etc.

Some commercial online survey programs are available, but none is properly designed for the whole process of school improvement survey. They are usually designed for the situation where a survey company already has a survey form (that may be written by the school or designed by the company) on its website. Then the school completes the form, and the data is directly transferred to the database in the server computer. Then the survey company analyzes the data, generates a report, and sends the report and raw data back to the school.

1.2   Purposes and Goal

This software "Smart School Improvement Online Survey System"(SSIOSS) is designed for school improvement surveys conducted by the Bureau of Education Service Applied Research (BESAR) in University of North Dakota. It will be used by system

administrator, school user, and end user (survey responders), following standard design principles and providing the following specific capabilities.

The project uses standard software engineering principles and techniques to design the program by applying artificial intelligence for on-line data collection for schools. This means the program we designed will function as a decision system to meet the requirements of the school improvement survey, and to standardize the data collection process for a school, make the evaluation process easier for the survey company and make the evaluation results (including data analysis output and report) more standard.

The program is designed to be intelligent and flexible. Users may choose to answer questions of local interest. It can automatically provide proper items from an item bank that is stored online for schools to choose from, depending on its own requirement. It can automatically generate a survey form online for a school containing the questions selected. It can provide recommendations and information for users to follow depending on the school's requirements. It can transfer data automatically between school and survey company, so that the time and money can be saved for school and survey company. Errors which often happen in the process of writing and data entry can be greatly reduced.

1.3 Structure of This Report

The next chapter describes the specification of the problem and the requirements of the project. Chapter 3 is focused on matching the requirement specification with the design of this project and using Class Diagrams of UML to show how theoretical Belief Desire Intention (BDI) are used in the whole project program, especially how to automatically generate a survey form for a school and the recommendation subsystem of

the project. Chapter 4 describes and presents each session of SSIOSS with an example on how school users used the system to work, how the system was used to terminate responding to survey questions, how the administrator used the system database to check the school's work in process and control the system, and how the recommendation program worked for the process of SSIOSS. Finally we conclude the report with ideas for future work and references.

Chapter II Requirements and Specifications

2.1 System requirements

The system will be used by three types of uses: (1) the system administrator who will manage the system. (2) The school user who represents the school to manage the NCA survey. (3) Survey user who answer the survey questions to the survey form.

Project deliverables will consist of (1) a packet of program: the source code. (2) a documentation: project specification, design, test, and user menu.

2.1.1 Server requirements

Server hardware and Operating System: personal computer with Windows 2000 or Windows 98 and should be connected to Internet daily.

Internet software: Java web server and Netscape Navigator or Microsoft Internet Explorer above 4.0.

Project language environment: a java compiler.

Database: Relational database- MS Access is selected, because of the prevalence in the current school system.

2.1.2 Client requirements

Client hardware and Operating System: Personal computer with currently is using operating systems.

Internet software: Netscape Navigator or Microsoft Internet Explorer above 4.0. Javascript enable.

2.2 Input and Output Requirements

Input is by keyboard or mouse from nice interface that is provided by a system either from client part computer (school user or survey user) or server part computer (system administrator).

Output is the html file or word file that is displayed on the screen as a response to the user's request.

2.3 Data Flow Diagrams

The project requirements are described by data flow diagrams [4] of relationship between each service function, each system components, and external user. First of all, the system structure is demonstrated in figure 1 and the relationship between system service and users is described in figure 2. The system help service is described in figure 3. The survey user service, school user services, and administrator services are presented in section 2.3.3, section 2.3.4, and section 2.3.5, respectively.

2.3.1   High-level structure of system diagram

When connected to the SSIOSS via a web connection, each different type of user interacts with the system through an appropriate menu - driven interface. The school user interface is only available to properly logged in users. Service requirement and data are transferred through servlet program to the decision-making subsystem and database.
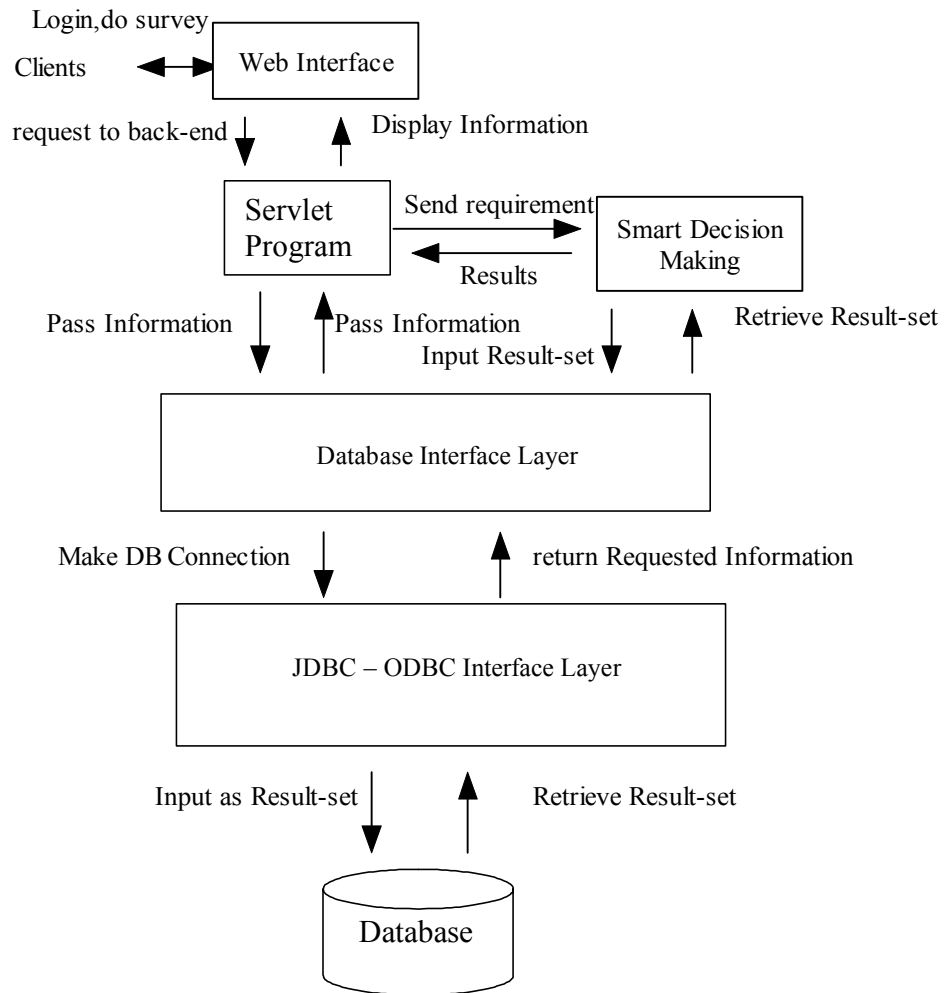
Figure 1 Architecture of SSIOSS
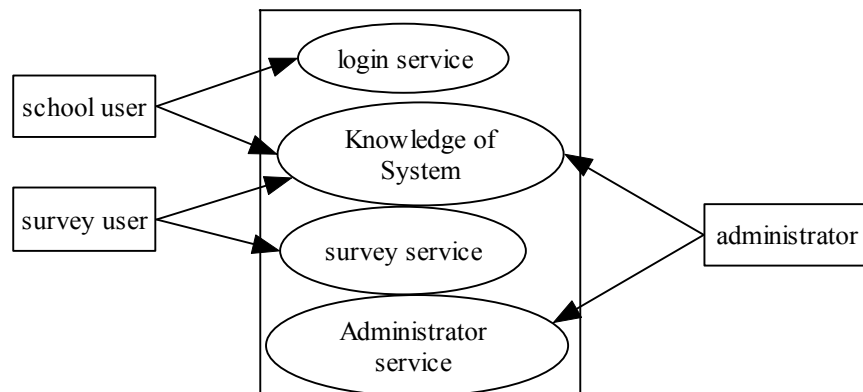
Figure 2 High level of system structure

2.3.2　Data flow diagram of the system help service

　　　　When connected to the SSIOSS via a web connection, school user, survey user

and administrator can view the user menu to know the current function of the system.

Also school user can get the needed direction via the recommendation service.
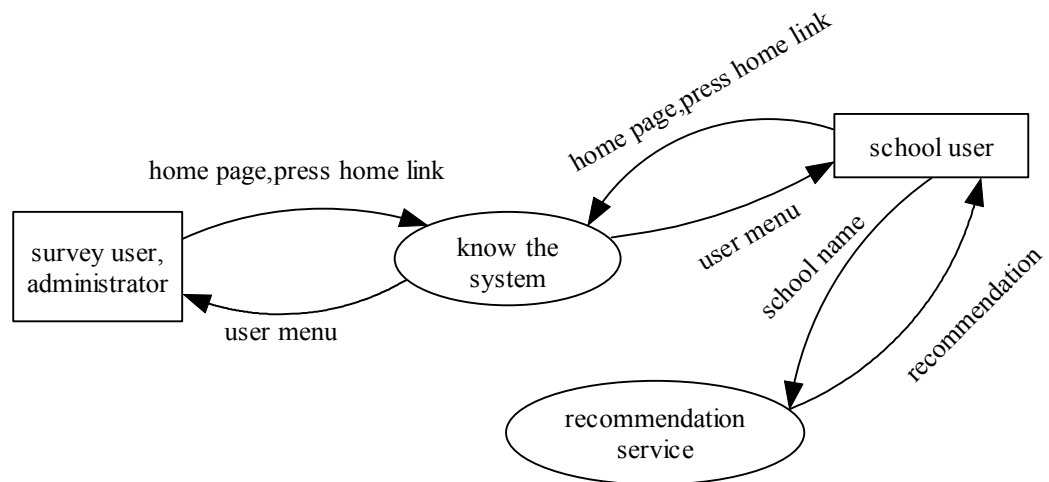
Figure 3 DFD of system help service

2.3.3　Survey user service

　　　　Survey users answer the questions in the survey form selected by school users.

Completed survey data is inserted to the database. Each different type of survey user may

answer different survey form, and the data will be inserted to corresponding database

table.
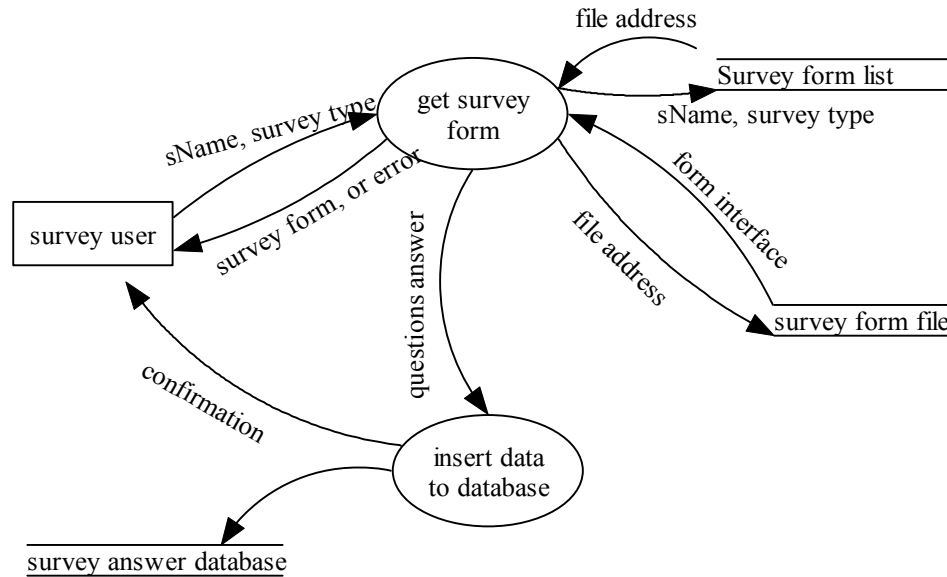
Figure 4 DFD of online survey service

2.3.4 School user service

The school user has two main jobs: 1) to create survey form for school, 2) to get the guideline from the system to make sure the survey form is appropriate to meet the requirement of the client [5]. Combination of these two is called Smart Decision Making System.

To create a survey form, a school user follows a certain sequence of steps. First, as a new user in a particular school survey system, the survey administrator needs to submit the information about the school and him/her self to the system. Second, he/she needs to submit the survey requirement to the system. Then, under the system guide, he or she can select items from the item bank and insert it to the database.

In the second part, the survey administrators from a school or school district can pull as many questions as needed to create a survey form they'd like to have according to their school accreditation own needs.

9

2.3.4.1 New school user registration

A new school user needs to create an account in SSIOSS. This process is called registration. Registration provides information of the school user itself and the school he/she presents, including user name and password that will be used in the future school improvement survey process.
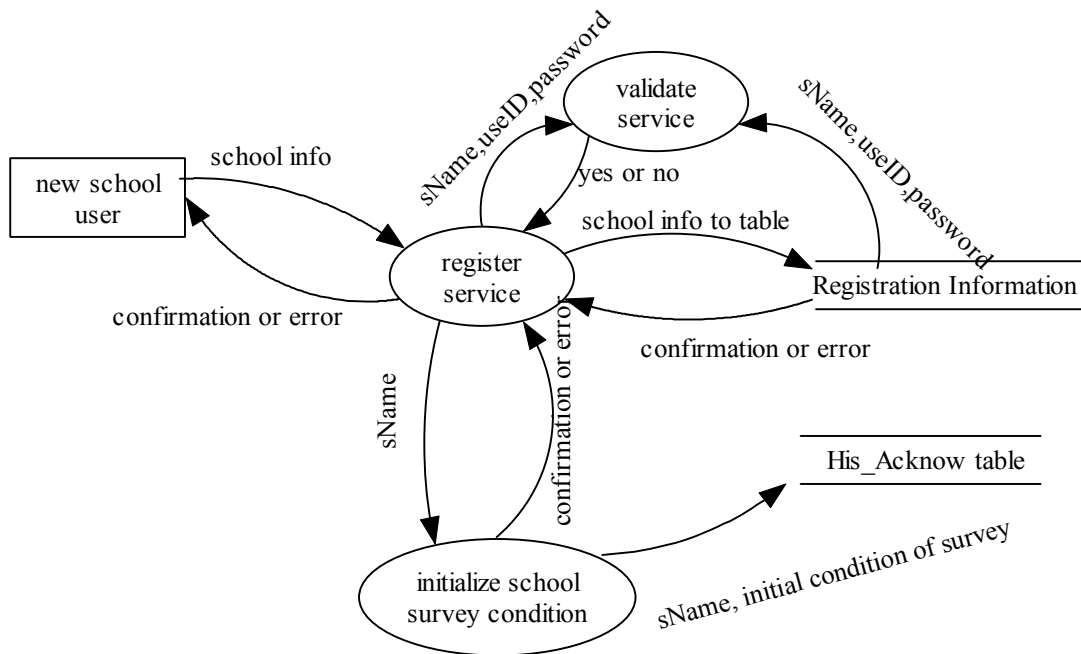
Figure 5 DFD of registration service

2.3.4.2 Log in service

School user must login correctly before they can access SSIOSS services. Successful login user is able to obtain the main service menu.
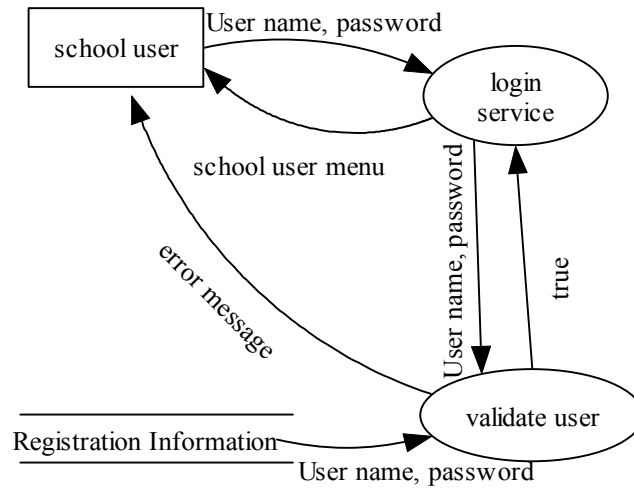
Figure 6 DFD of login service

2.3.4.3 Submit survey requirement
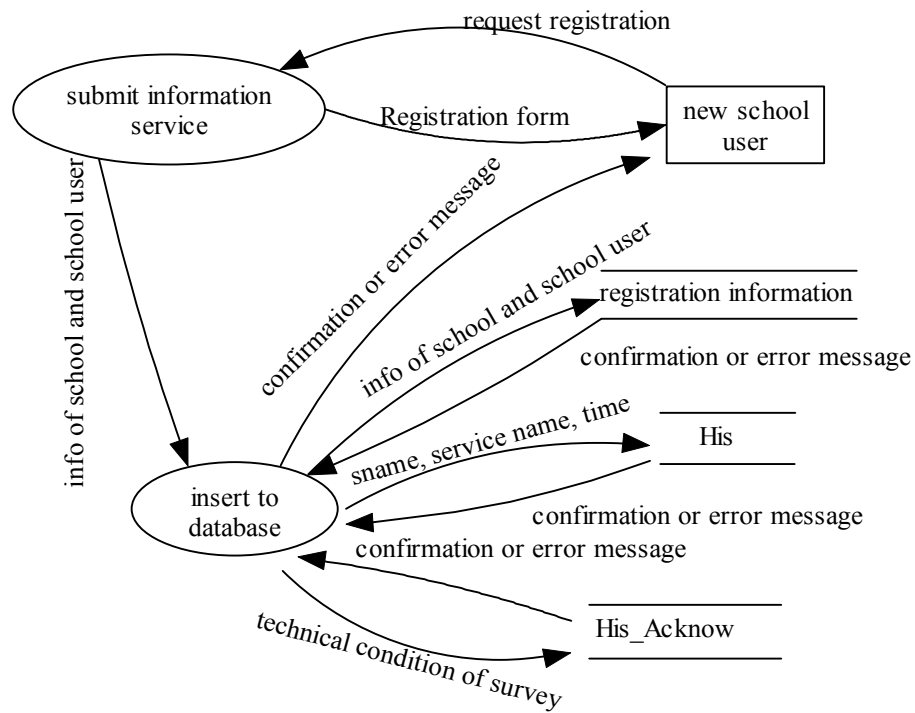
Figure 7 DFD of submit requirement

2.3.4.4 Data flow diagram of select question
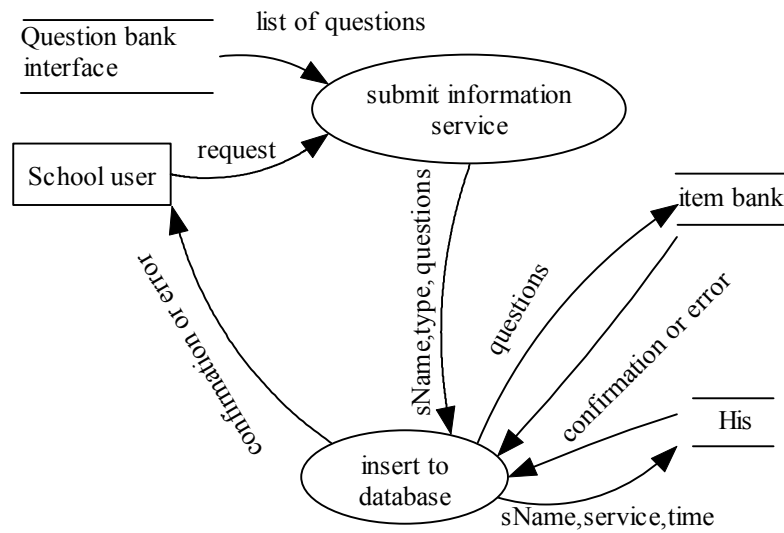
Figure 8 DFD of select question

2.3.4.5 Data flow diagram of create survey form
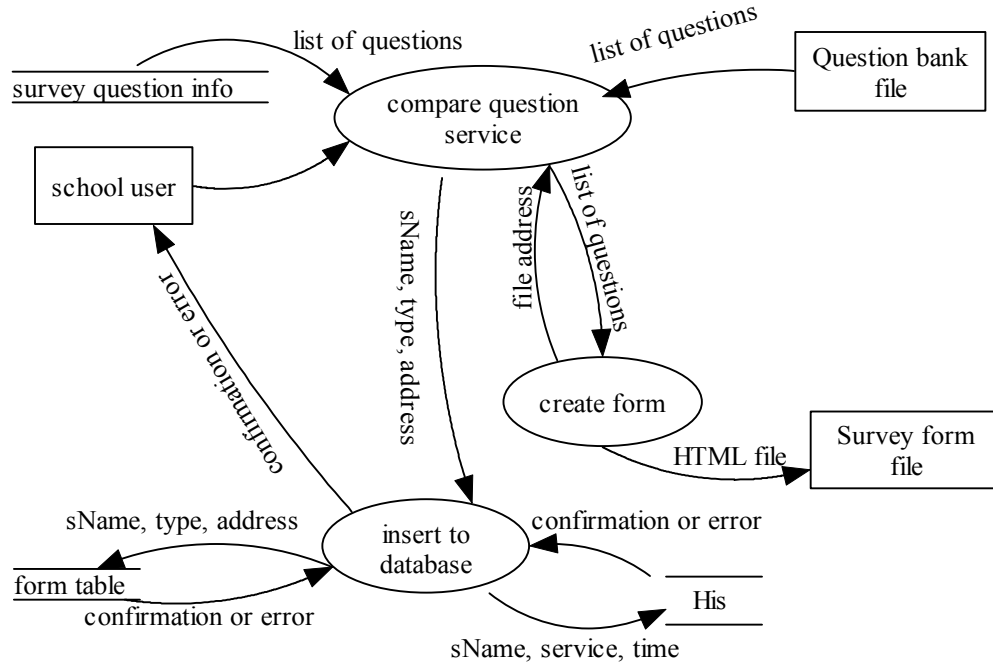
Figure 9 DFD of create survey form

2.3.4.6 Price status

This procedure provides information such as how long a school improvement survey will be taken and how much money a user school needs to pay for a survey. The amount of time and money charged for a survey very much depends on the specific requirements. After a school user obtains the information, they can change the requirement to adjust to the amount of money and time needed for the task.
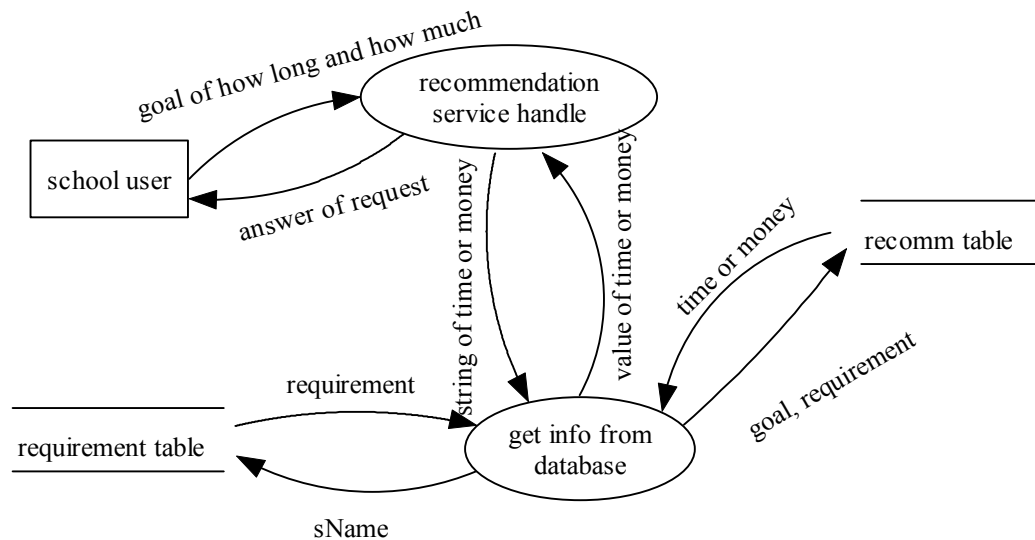
Figure 10 DFD of get recommendation of time and money

2.3.4.7 Direction guide

This procedure helps school users when they are new for the school improvement survey. It gives directions to school users of what should be done for the next step.
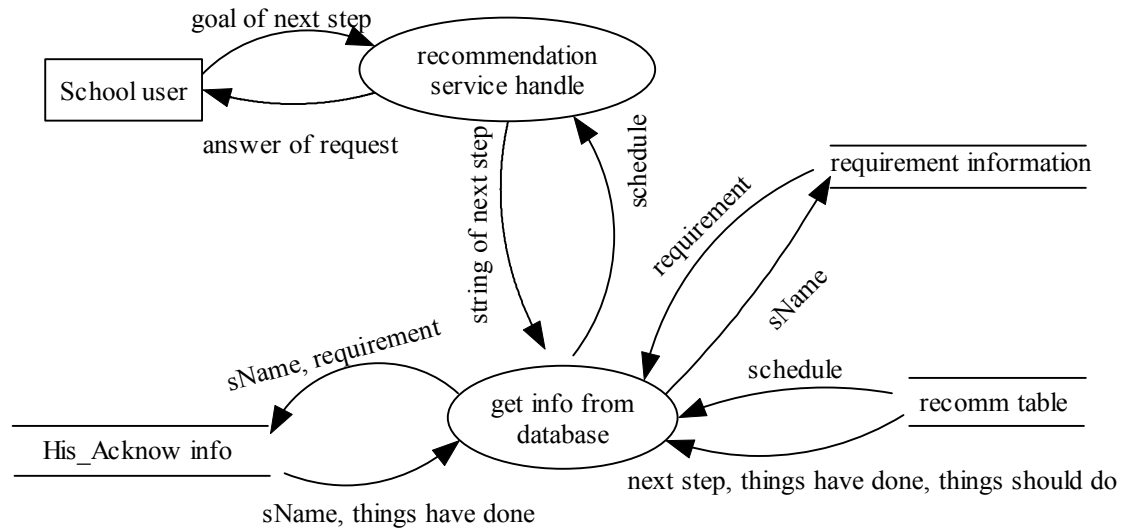
Figure 11 DFD of get recommendation of next step

2.3.5   Administrator service

The main job of administrator using server side computer is to manage the

SSIOSS system, adding items to the item bank, delete outdated items, delete files (tables,

html files), check school users, and maintain system service, etc.

2.3.5.1 Data flow diagram of delete file
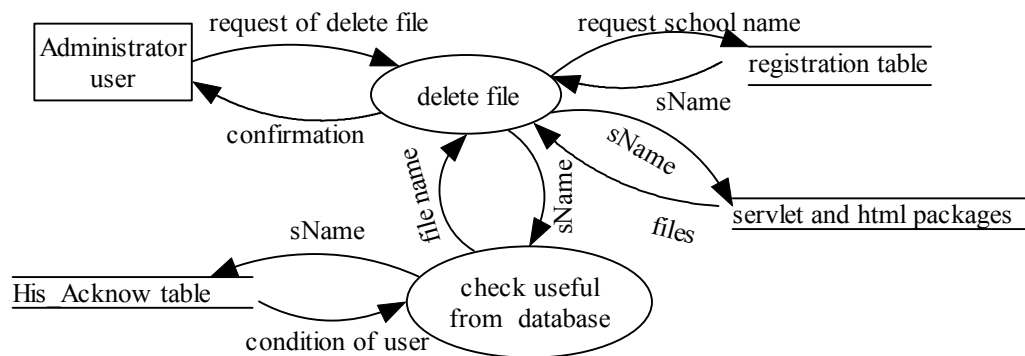
Figure 12 DFD of delete file

14

2.3.5.2 Data flow diagram of check school user
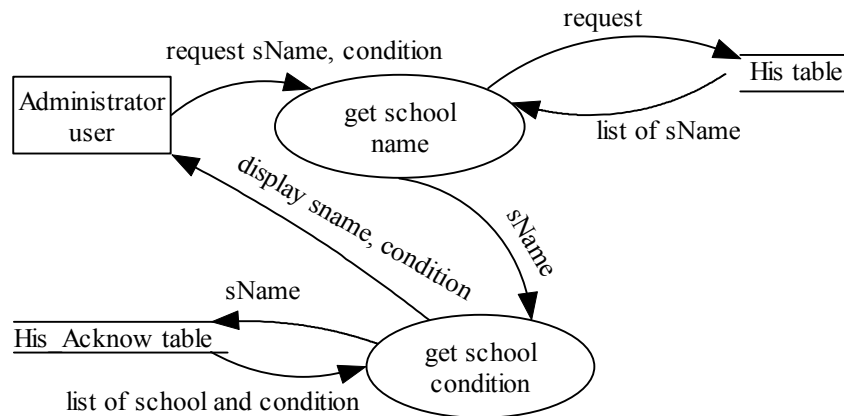
Figure 13 DFD of check school name and status

2.4 Entities and relationships

The system uses Access Database named SSIOSS.mdb. SSIOSS contains 9 basic tables, and will automatically increases in size while a school user works with the system. They are history, form, His_Acknow, ItemBank1, ItemBank2, recomm, registration, report, requirement, and User data table. The last one is automatically created to store survey data for each type of survey in school. User data table is a general name and temporarily used here, because a real table has not been created yet. The history table is used to keep record of each job a school user has done. The form table is used to store the address of each form that school user has created. The His_Acknow table is used to keep technical information that will be used in the process of creating a survey form, program, and user data table. The ItemBank1 and ItemBank2 are used to store questions that the school user selected. The recomm table is used to store information of the time, money, and next step. This information is plans that are created by BESAR based on the survey experience. The registration table is used to keep record of registration information for each school and school user. The report table is used to store the address of the reports

15

that the data analyst generated. The requirement table is used to store information of survey requirement for each school. The relationships among each of the basic tables are presented in Figure 14 [3].
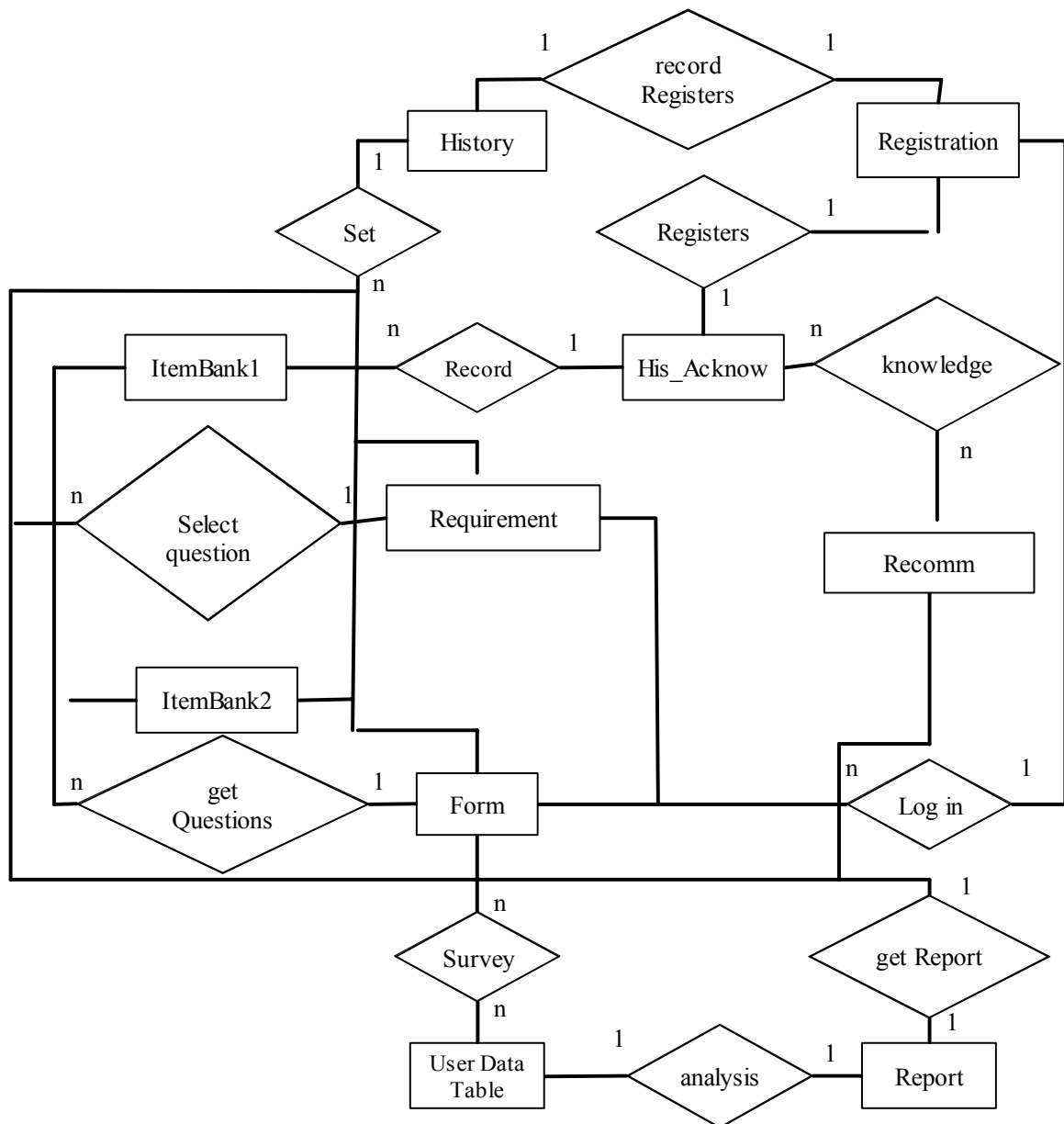
Figure 14 Entities and Relationship of SSIOSS Database

2.5 Project Process

Incremental development was used for this project. It includes three increments. They are simple school improvement online survey, adding items to a survey form and generating survey form, and adding smart decision-making system. Because each increment is almost independent. After each increment is projected, a deliverable increment of the software is generated. But each of increment is more intelligent and flexible than the last one.

The first increment is an example of school improvement online survey. A java environment was set up to make the data source of database available on the MS Windows 2000, to create table of survey information, and to manually create survey form. Each component, integrated and connected each component was implemented. Then the whole system was debugged and tested.

The second increment is to add functionalities of item bank to automatically generate a survey form online. Depending on the educational measurement expertise, BESAR developed items (from many areas and perspectives regarding school improvement issues) to form an item bank for school improvement survey purpose. These survey items were posted online. Program was projected to automatically generate a survey form. New tables were created to store the survey form. Java servlet program was applied to insert, update each column of survey form table. We compile the adding part and integrate the system. After this increment, the system is able to generate and process data collection and analysis automatically, thus time and money efficient because they don't need to meet face to face, discussing the items of the survey form any more.

This is especially applicable when the school and the survey company are not in the same city.

The third increment is to add powerful functionality of SDMS to the system. This allows a survey company to provide the service of SSIOSS based on the experience and requirement of survey from school. After this increment, the system is totally more upgraded than the previous increment.

CHAPTER III: Software Design and Implementation

3.1 The Architecture of Smart School Improvement Online Survey System

The methodology applied in the project was to develop an interactive system integrating three tiers. Depending on project requirement specification, we follow top-down design concept and implement the corresponding high-level architecture first. This concept is illustrated in Figure 15 below.
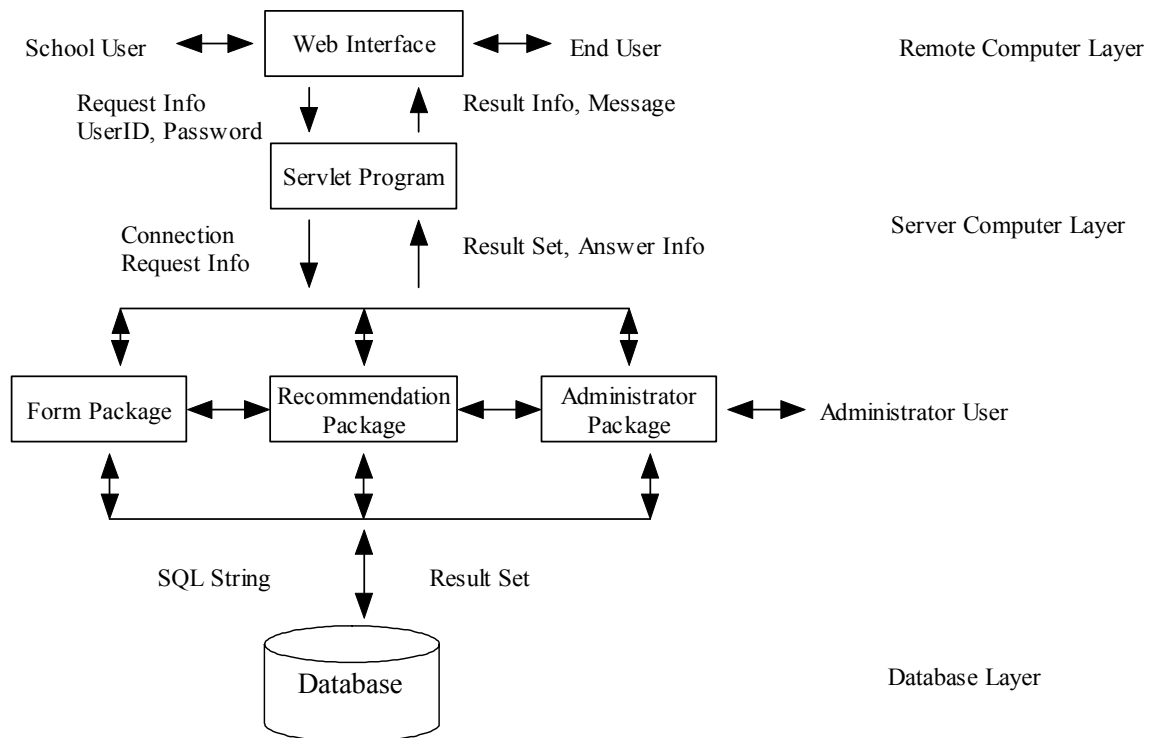
Figure 15 Architecture of SSIOSS System

3.2 UML Design for Back-End Program

In this section we use Unified Modeling Language (UML) notation and Object – Oriented (OO) design techniques to present the design of SSIOSS system [6].

3.2.1   Package Construct

In Figure 16, we present a package view of the SSIOSS system. There are three main parts, 1) Interface Package, 2) SSIOSS Database, and 3) Servlets Package.   The Interface and Database Packages are described in greater detail in section 3.3. In this section, the Servlet package, a back–end program is described. The Servlet package, illustrated in Figure 17, contains three sub-packages (Form, Recomm and Administrator) which are illustrated in Figures 18-20.
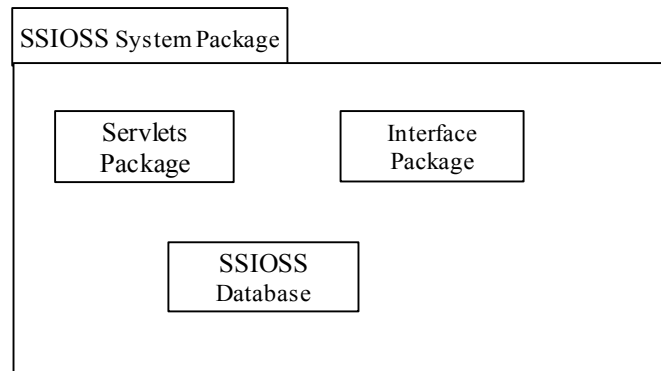
Figure 16 Package construct of SSIOSS System

From Figure 17 Package Construct of Servlets Package, we can see that it is a whole back-end package. It was also composed of Form package, Administrator package, Recomm Package, and ten classes.   There are class diagrams for these classes in section 3.2.3.
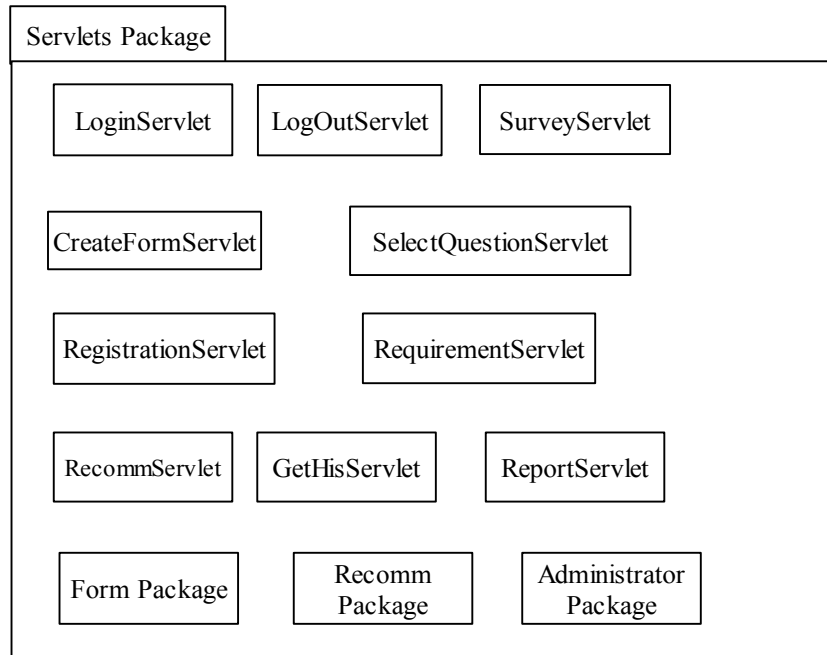
Figure 17 Package construct of Servlets Package
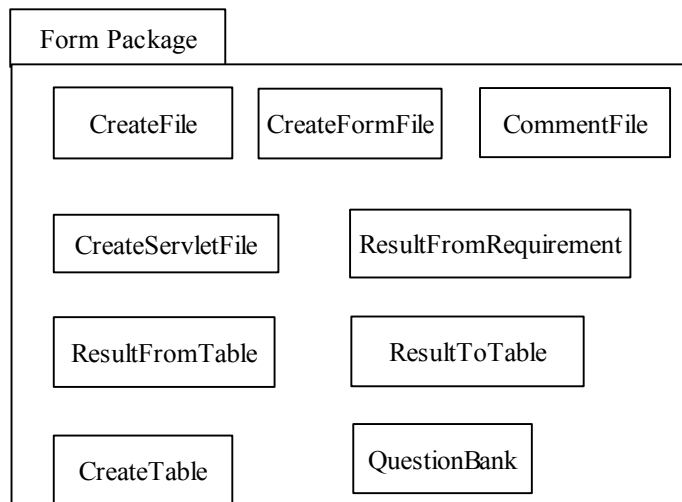
Figure 18 Package construct of Form Package
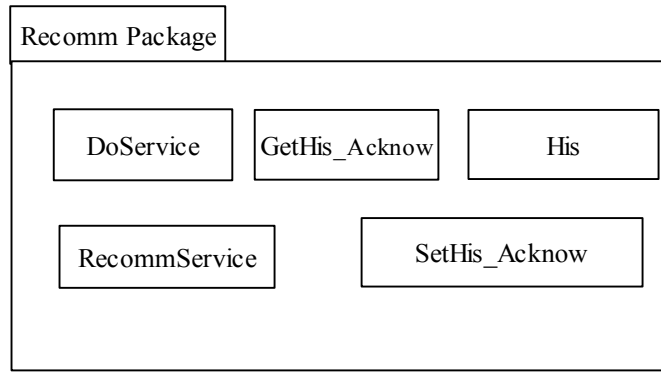
Figure 19 Package construct of Recomm Package
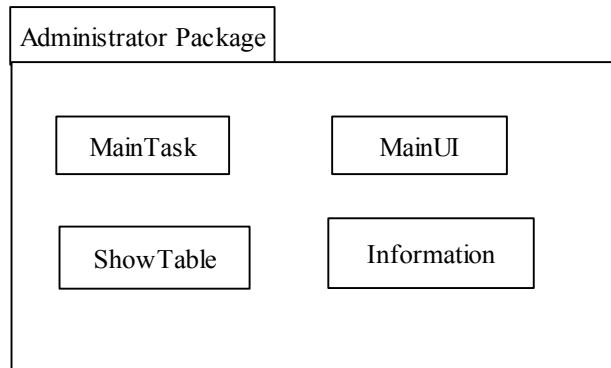
Figure 20 Package construct of Administrator Package

3.2.2 Use Case Diagram

In this section, we use the Use Case Diagram to demonstrate user and program relationship. Figure 21 presents Package level Use Case Diagram of SSIOSS. Figure 22 presents End User class level Use Case Diagram of SSIOSS. Figure 23 describes Administrator class level Use Case Diagram of SSIOSS. The class level of Use Case Diagram for school user is similar to Figure 21 but is augmented by adding classes to the package.
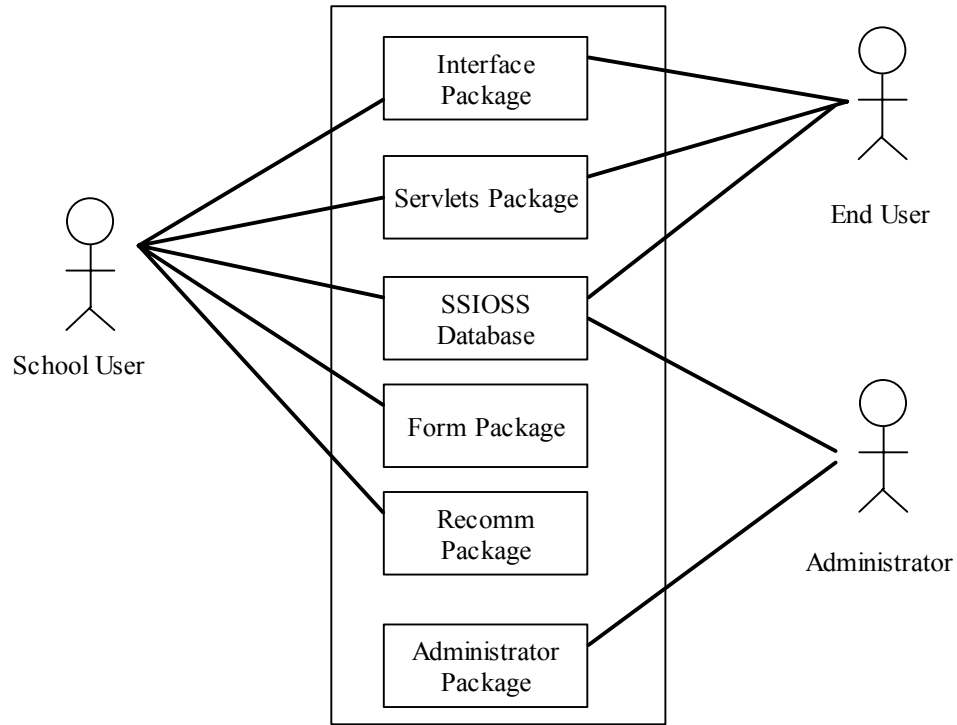
Figure 21 Package level Use Case Diagram of SSIOSS

In Figure 22, we list the html files that are related to End User in the Interface Package and the java classes that are related to End User in servlets Package. School_people_surveyform.html, school_people_servlet.java, and school_people table are automatically generated while School User is working with SSIOSS system.

Figure 23 lists the java classes and tables that are related to Administrator operation procedures.
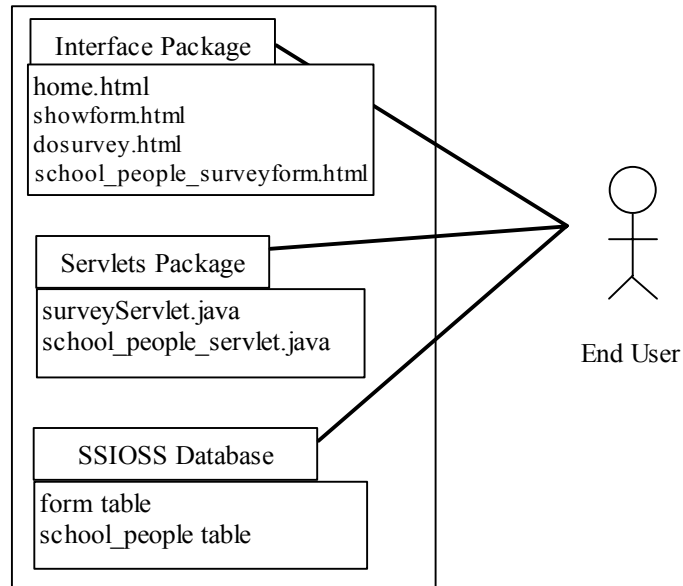
Figure 22 End User class level Use Case Diagram of SSIOSS
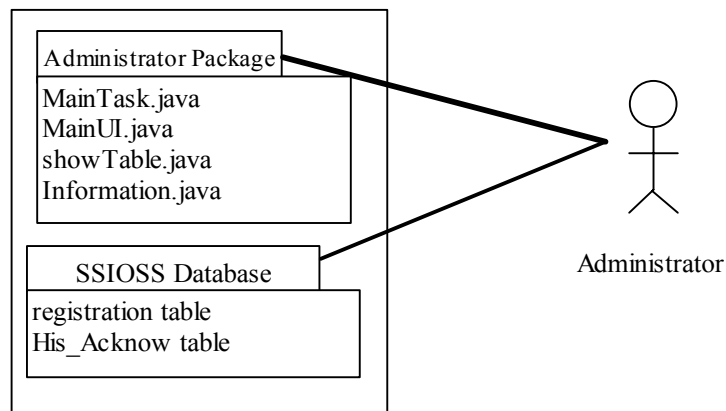
Figure 23 Administrator class level Use Case Diagram of SSIOSS

3.2.3 Class Diagram of SSIOSS

There are 30 classes in the SSIOSS. So each Class Diagram concentrates on the center class constructs and relationships between classes. The center class construct is focused on. Other related classes are detailed late. Figures 24 to 32 describe the servlet classes. These classes are related with front - end of web pages used by school user or end user and back-end program in the Form and Recomm packages. Figures 33 to 37

24

describe the classes of recomm package and the relationships between them [1]. Figure 38 to 46 describe classes of the form package and relationships between them. Figure 47 to 50 describe administrator classes.

HttpServlet

(Specialization)

LoginServlet
connection: Connction=null statement: Statement=null URL:String ="jdbc:odbc:SSIOSS"

init(config);
doPost(HttpServletRequest,HttpServletResponse);
destroy();
getServletInfo():String;

(uses)

(uses)

Login.html

Registration Table
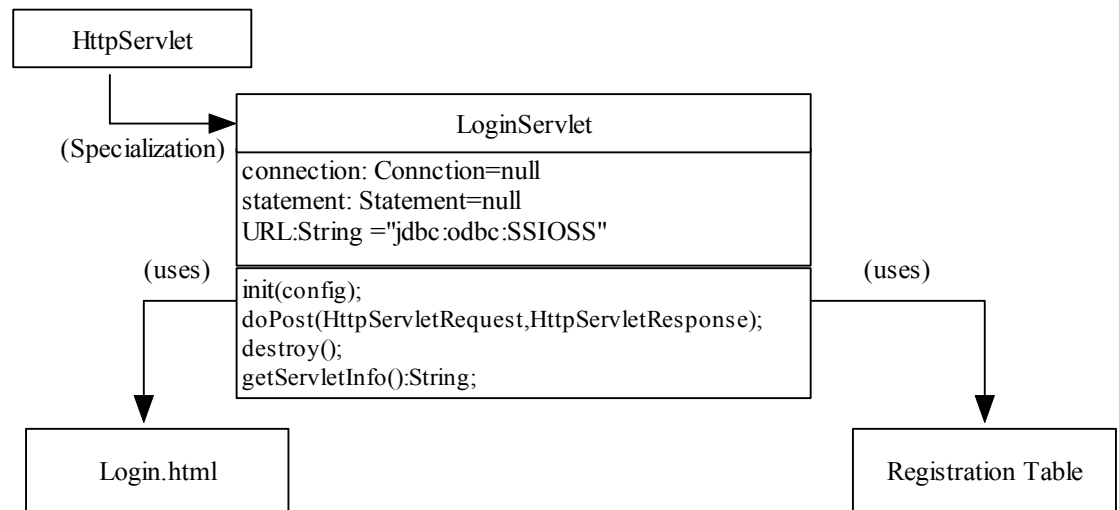
Figure 24 LoginServlet Class Diagram

HttpServlet

(Specialization)

LogOutServlet
doGet(HttpServletRequest,HttpServletResponse); getServletInfo():String;
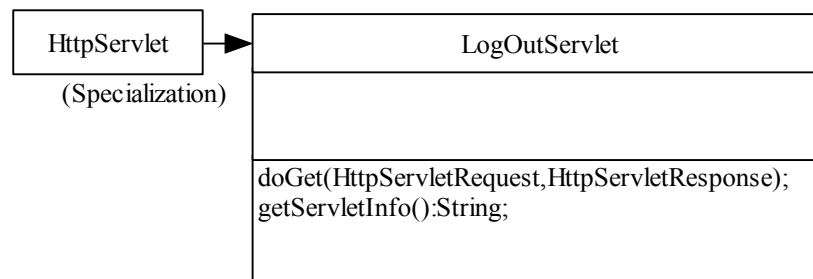
Figure 25 LogOutServlet Class Diagram

```
┌──────────────┐      ┌───────────────────────────────────────────────────────┐
│ HttpServlet  │─────▶│                 RegistrationServlet                    │
└──────────────┘      ├───────────────────────────────────────────────────────┤
  (Specialization)    │ connection: Connction=null                             │
                      │ statement: Statement=null                              │
                      │ URL:String ="jdbc:odbc:SSIOSS"                          │
                      ├───────────────────────────────────────────────────────┤
                      │ init(config);                                          │
                      │ doPost(HttpServletRequest,HttpServletResponse);        │
                      │ destroy();                                             │
                      │ insertIntoDB():boolean;                                │
                      │ setHis_Acknow.setRegistration(String);                 │
                      │ His.setHis(String);                                    │
                      └───────────────────────────────────────────────────────┘
```

(uses)   (uses)

```
┌───────────────┐  ┌────────────────────┐    ┌──────────────────┐   ┌──────────────┐
│ Register.html │  │ Registration Table │    │  setHis_Acknow   │   │     His      │
└───────────────┘  └────────────────────┘    └──────────────────┘   └──────────────┘
```
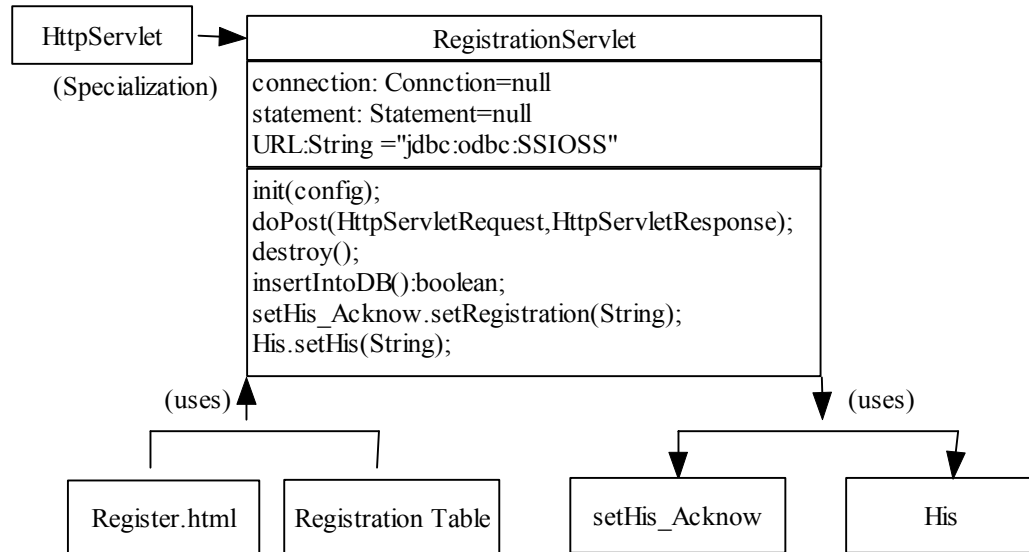
Figure 26 RegistrationServlet Class Diagram

```
┌──────────────┐      ┌───────────────────────────────────────────────────────┐
│ HttpServlet  │─────▶│                     SurveyServlet                      │
└──────────────┘      ├───────────────────────────────────────────────────────┤
  (Specialization)    │ connection: Connction=null                             │
                      │ statement: Statement=null                              │
                      │ URL:String ="jdbc:odbc:SSIOSS"                          │
                      ├───────────────────────────────────────────────────────┤
                      │ init(config);                                          │
                      │ doPost(HttpServletRequest,HttpServletResponse);        │
                      │ destroy();                                             │
                      │ insertIntoDB():boolean;                                │
                      │ school_peopleServlet.doPost(HttpServletRequest,        │
                      │ HttpServletResponse);                                  │
                      └───────────────────────────────────────────────────────┘
```

(uses)   (uses)

```
┌──────────────────────┬───────────────┐   ┌────────────┐   ┌──────────────────────────┐
│ school_people_Servlet│ doSurvey.html │   │ Form Table │   │ school_people_survey Table│
└──────────────────────┴───────────────┘   └────────────┘   └──────────────────────────┘
```

Figure 27 SurveyServlet Class Diagram
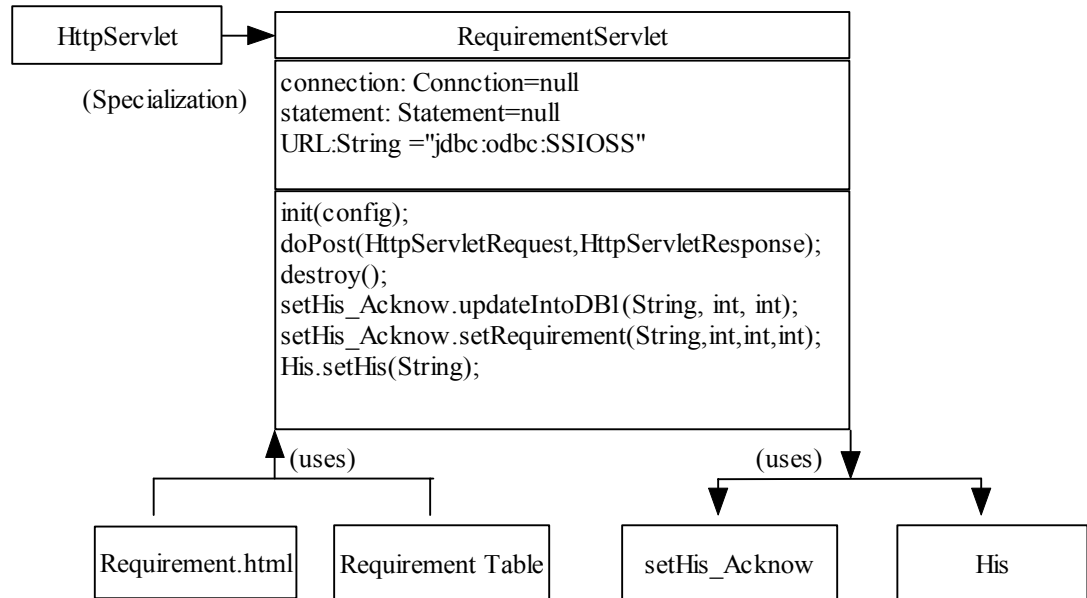
Figure 28 RequirementServlet Class Diagram
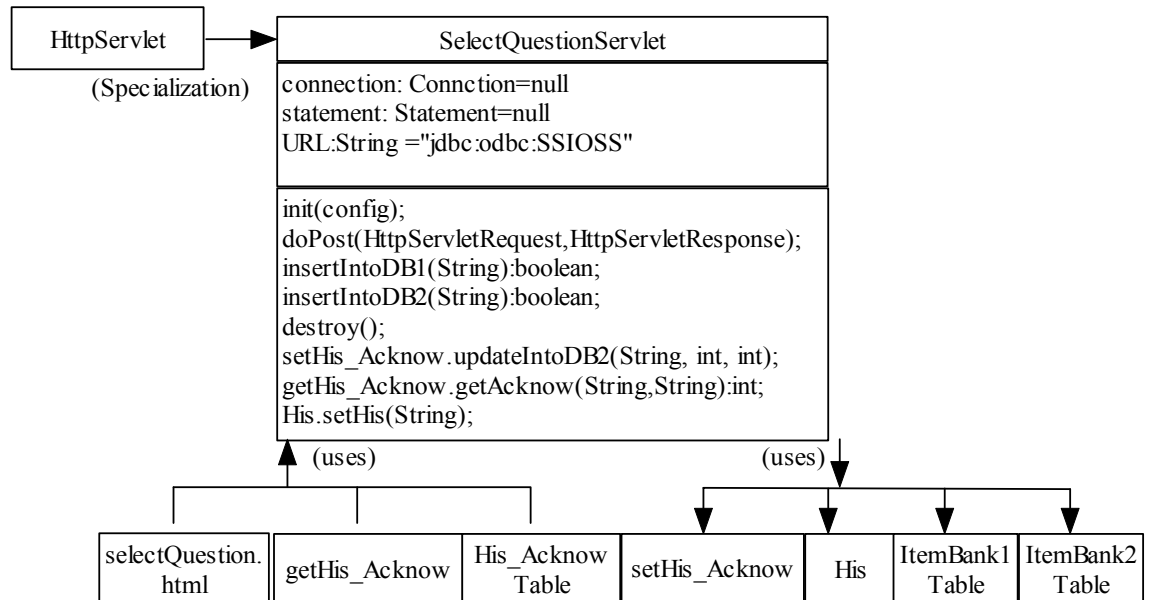
Figure 29 SelectQuestionServlet Class Diagram
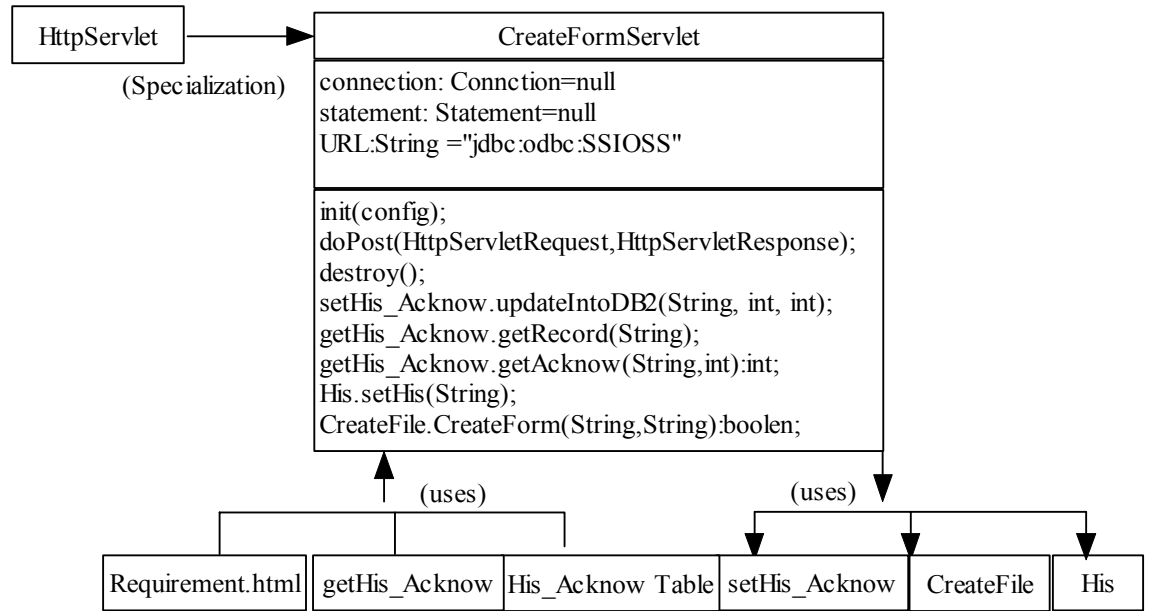
Figure 30 CreateFormServlet Class Diagram
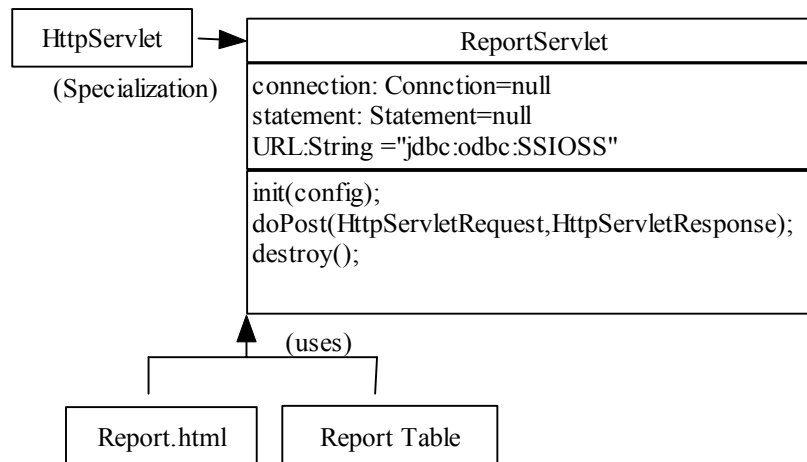
Figure 31 ReportServlet Class Diagram
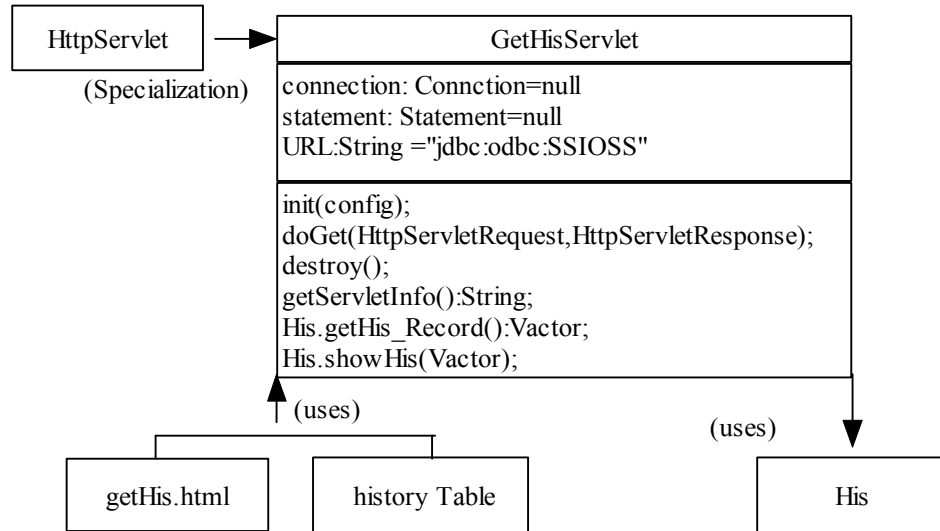
Figure 32 GetHisServlet Class Diagram
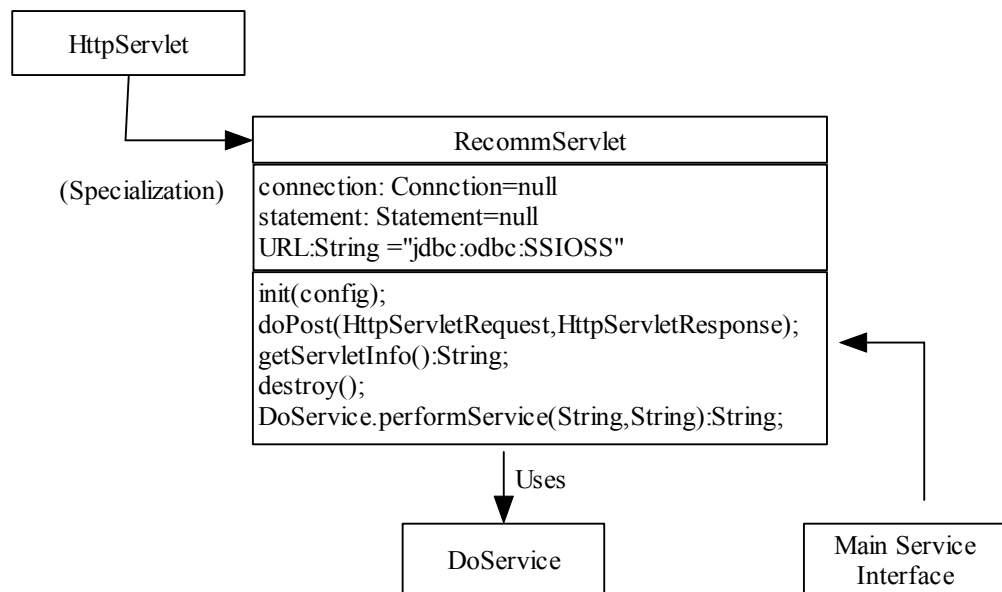
Figure 33 RecommServlet Class Diagram

```
                    ┌────────────────────────────────────────┐
                    │              DoService                  │
                    ├────────────────────────────────────────┤
                    │                                         │
                    ├────────────────────────────────────────┤
                    │ DoService(Connection);                  │
                    │ performService(String,String):String;   │
                    │ GetHis_Acknow.getRecord(String):boolean; │
                    │ GetHis_Acknow.getAcknow(String):int;     │
                    │ RecommService.getPlan(String,int):String;│
                    └────────────────────────────────────────┘
```

Uses   Uses

```
┌──────────────┐   ┌──────────────┐   ┌──────────────┐
│ GetHis_Acknow│   │ RecommServlet│   │ RecommService│
└──────────────┘   └──────────────┘   └──────────────┘
```
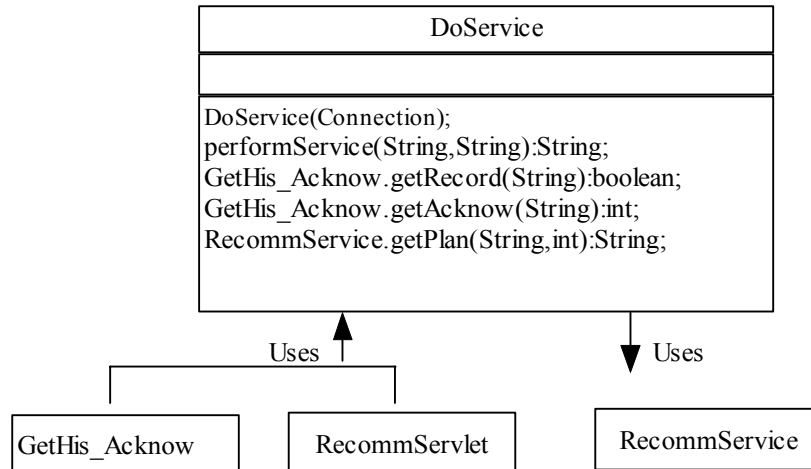
Figure 34 DoService Class Diagram

```
┌──────────────────────────────────────┐        ┌──────────────┐
│            GetHis_Acknow              │◄──(uses)│ His_Acknow   │
├──────────────────────────────────────┤        │   Table      │
│ connection: Connction                 │        └──────────────┘
│ caknow:int[]                          │
├──────────────────────────────────────┤
│ GetHis_Acknow(Connection)             │
│ getRecord(String):boolean;            │
│ getAcknow(String):int;                │
│ getAcknow(String,String):int;         │
│                                       │
└──────────────────────────────────────┘
```
▲ (uses)
```
┌──────────────┐
│ His_Acknow   │
│   Table      │
└──────────────┘
```
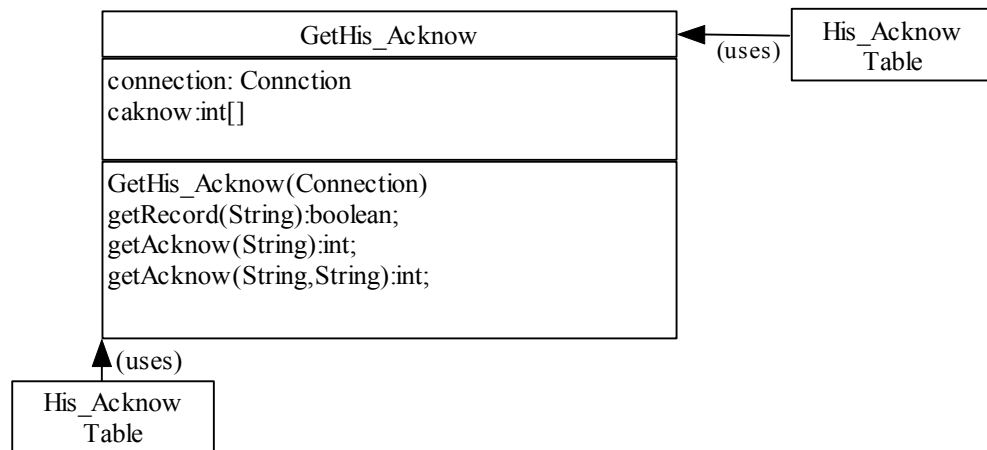
Figure 35 GetHis_Acknow Class Diagram

(uses)
```
┌──────────────────────────────────────┐        ┌──────────────┐
│            RecommService              │◄───────│  DoService   │
├──────────────────────────────────────┤        └──────────────┘
│ connection: Connction                 │
│                                       │
├──────────────────────────────────────┤
│ RecommService(Connection)             │
│ getPlan(String,int):String;           │
│                                       │
└──────────────────────────────────────┘
```
▲ (uses)
```
┌──────────────┐
│ recomm Table │
└──────────────┘
```
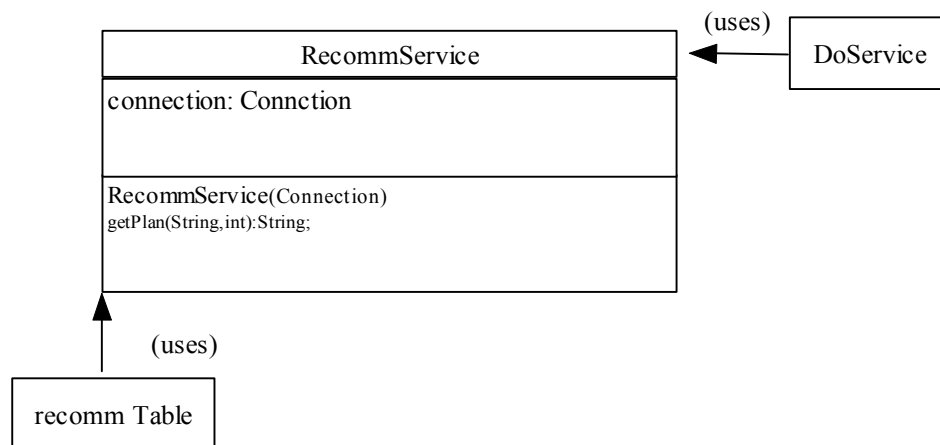
Figure 36 RecommService Class Diagram

```
┌─────────────────────────────────────────────┐                    ┌──────────────┐
│              SetHis_Acknow                    │◄───────────────────│  DoService   │
├─────────────────────────────────────────────┤        (uses)      └──────────────┘
│ connection: Connction                         │
│ sName:String                                  │
│ regis,require,SQ1,SQ2,SQ3,SQ4,                │
│ SF1,SF2,SF3,SF4,                              │
│ ST1,ST2,ST3,ST4,                              │
│ report:int                                    │
│                                               │
├─────────────────────────────────────────────┤
│ SetHis_Acknow(Connection)                     │
│ setRegistration(String);                      │
│ setRequirement(String,int,int,int,int);       │
│ insertIntoDB(String):boolean;                 │
│ updateIntoDB1(String,int,int):boolean;        │
│ updateIntoDB1(String,String,String):boolean;  │
└─────────────────────────────────────────────┘
```

(uses)

```
┌──────────────┐
│ His_Acknow   │
│   Table      │
└──────────────┘
```

Figure 37 SetHis_Acknow Class Diagram

```
┌─────────────────────────────────────────────┐                    ┌──────────────┐
│                    His                        │◄───────────────────│  DoService   │
├─────────────────────────────────────────────┤                    └──────────────┘
│ connection: Connction                         │        (uses)
│ sName:String                                  │                    ┌──────────────┐
│ workTime:String[]                             │                    │ GetHisServlet│
│ purpose:String[]                              │                    └──────────────┘
├─────────────────────────────────────────────┤
│ His(Connection,String)                        │
│ setHis(String);                               │
│ insertIntoDB(String):boolean;                 │
│ getHis_Record1():Vector;                      │
│ getNextRow(ResultSet,ResultSetMetadata):Vector;│
│ showHis(Vector);                              │
└─────────────────────────────────────────────┘
```

(uses)

```
┌──────────────┐
│ history Table│
└──────────────┘
```

Figure 38 His Class Diagram

31

Figure 39 CreateFile Class Diagram

Figure 40 QuestionBank Class Diagram

```
┌─────────────────────────────────────────────┐                    ┌─────────────────┐
│                CreateFormFile                 │◄───────────────    │                 │
├─────────────────────────────────────────────┤      (uses)        │   CreateFile    │
│                                               │                    │                 │
│                                               │                    └─────────────────┘
├─────────────────────────────────────────────┤
│ makeSurveyForm(String[],RandomAccessFile,int):│
│ boolean;                                      │
│                                               │
└─────────────────────────────────────────────┘
```

Figure 41 CreateFormFile Class Diagram

```
┌─────────────────────────────────────────────┐                    ┌─────────────────┐
│               CreateServletFile               │◄───────────────    │                 │
├─────────────────────────────────────────────┤      (uses)        │   CreateFile    │
│                                               │                    │                 │
│                                               │                    └─────────────────┘
├─────────────────────────────────────────────┤
│ makeServletFile(String[],RandomAccessFile,int):│
│ boolean;                                      │
│ CommentFile.writeToFile(String,String,String,String);│
│                                               │
└─────────────────────────────────────────────┘
                      │
                      ▼
             ┌─────────────────┐
             │   CommentFile   │
             └─────────────────┘
```

Figure 42 CreateServletFile Class Diagram

```
┌─────────────────────────────────────────────┐                    ┌─────────────────┐
│                 CreateTable                   │◄───────────────    │                 │
├─────────────────────────────────────────────┤      (uses)        │   CreateFile    │
│ connection:Connection                         │                    │                 │
│                                               │                    └─────────────────┘
├─────────────────────────────────────────────┤
│ CreateTable(Connection);                      │
│ setTable(String,String[],int);                │
│ createTableInDB(String):boolean;              │
│                                               │
│                                               │
└─────────────────────────────────────────────┘
```

Figure 43 CreateTable Class Diagram

```
CommentFile                              ◄─── CreateServletFile
                              (uses)
RandomAccessFile

writeToFile(String,String,String,String);
openFile(String);
closeFile(RandomAccessFile);
insertIntoFile(String,RandomAccessFile);
```

Figure 44 CommentFile Class Diagram

```
ResultFromRequirement                    ◄─── CreateFile
                              (uses)
connection:Connection
result:int[]
myresult:String[]
currentIndex:int

ResultFromRequirement(Connection);
getRecord(String);
getInfo(String);

                  ▲
           requirement
              Table
```
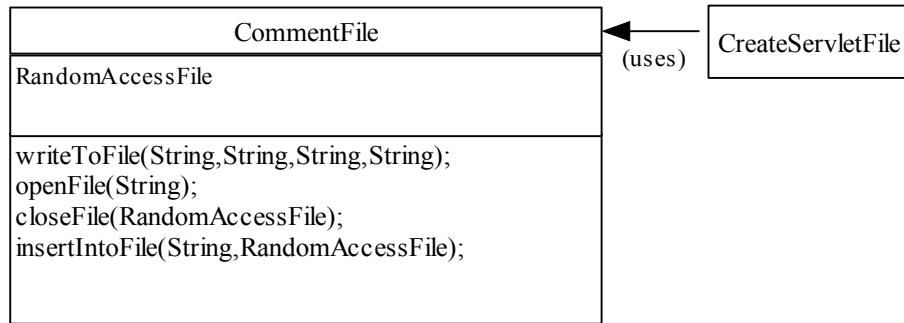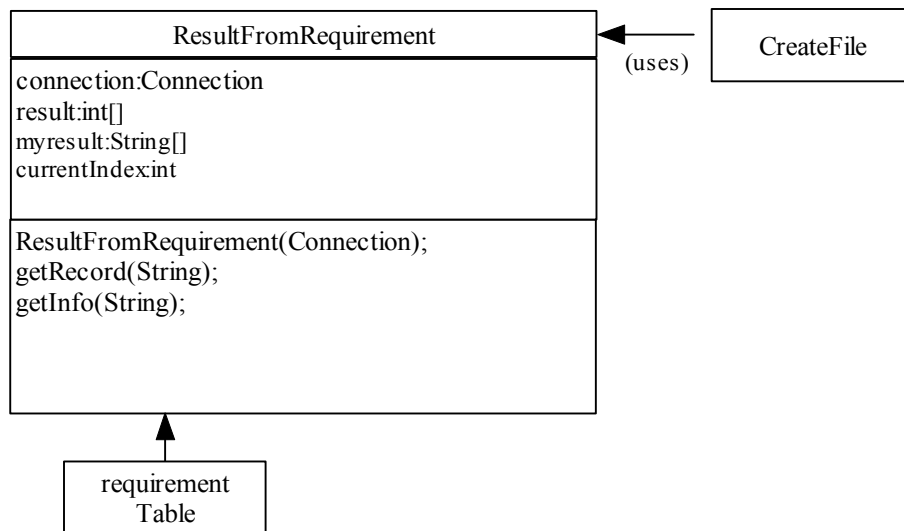
Figure 45 ResultFromRequirement Class Diagram

```
ResultFromTable                          ◄─── CreateFile
                              (uses)
connection:Connection
result:int[]
result1:int[]
result2:int[]

ResultFromTable(Connection);
getRecord(String,String);

                  ▲
         ┌────────┴────────┐
   ItemBank1 Table    ItemBank2 Table
```

Figure 46 ResultFromTable Class Diagram

```
┌─────────────────────────────────────────┐                ┌──────────────┐
│              ResultToTable               │◄───────────────│  CreateFile  │
├─────────────────────────────────────────┤   (uses )      └──────────────┘
│ connection:Connection                    │
│ userID:String                            │
│                                          │
├─────────────────────────────────────────┤
│ ResultToTable(Connection);               │
│ setRecord(String,String,String);         │
│ insertIntoDB(String);                    │
└─────────────────────────────────────────┘
                    ▲
                    │  (uses )
            ┌───────────────┐
            │  form Table   │
            └───────────────┘
```

Figure 47 ResultToTable Class Diagram

```
┌───────────────────────────────────────────────────┐        ┌──────────┐
│                     MainTask                        │◄───────│  JFrame  │
├───────────────────────────────────────────────────┤        └──────────┘
│ ListSchool, ListProcess,DeleteFile,Cancel,Open:JButton│     (Specialization)
│ connection:Connection                               │
│ URL:String                                          │
│ SN[]:String                                         │
│                                                     │
├───────────────────────────────────────────────────┤
│ MainTask( );                                        │
│ deleteFile():boolean;                               │
│ getSchool();                                        │
│ openFile();                                         │
│ schoolList();                                       │
│ shotDown();                                         │
│ Main();                                             │
│ MainUI.getDoTask1():JButton;                        │
│ MainUI.getDoTask2():JButton;                        │
│ MainUI.getDoTask3():JButton;                        │
│ MainUI.getDoTask4():JButton;                        │
│ MainUI.getDoTask5():JButton;                        │
│ MainUI.setFieldValue(String);                       │
│ MainUI.displaySchool(String[]);                     │
│ ShowTable.schoolName(ResultSet):String[];           │
│ ShowTable.displayResultSet(ResultSet);              │
│                                                     │
└───────────────────────────────────────────────────┘
            │         (uses)              ▲
            │                             │  (uses)
   ┌────────┼────────┐                    │
   ▼        ▼        ▼                    │
┌──────┐ ┌─────────┐ ┌───────────┐ ┌─────────────┐
│MainUI│ │ShowTable│ │Information│ │ StartAdminis│
└──────┘ └─────────┘ └───────────┘ └─────────────┘
```

Figure 48 MainTask Class Diagram

35

```
  ┌─────────┐            ┌──────────────────────────────────────────┐            ┌───────────┐
  │ JPanel  │──────────▶ │                 MainUI                   │ ◀──────────│  MainTask │
  └─────────┘            ├──────────────────────────────────────────┤            └───────────┘
                         │ field:JTextArea                          │              (uses)
  (specialization)       │ ListSchool,ListProcess,DeleteFile,       │
                         │ Canel,Open:JButton                       │
                         │ innerPanelSouth,innerPanelCenter:JPanel  │
                         │                                          │
                         ├──────────────────────────────────────────┤
                         │ MainUI();                                │
                         │ displaySchool(String);                   │
                         │ setFieldValue(String);                   │
                         │ getFieldValue():String;                  │
                         │ clearFields();                           │
                         │ getDoTask1():JButton;                    │
                         │ getDoTask2():JButton;                    │
                         │ getDoTask3():JButton;                    │
                         │ getDoTask4():JButton;                    │
                         │ getDoTask5():JButton;                    │
                         └──────────────────────────────────────────┘
```

Figure 49 MainUI Class Diagram

```
  ┌──────────────────────────────────────────┐            ┌───────────┐
  │               Information                 │ ◀──────────│  MainTask │
  ├──────────────────────────────────────────┤            └───────────┘
  │ connection:Connection                    │              (uses)
  │                                          │
  ├──────────────────────────────────────────┤
  │ Information(Connection);                  │
  │ getSchool():String[];                     │
  │                                          │
  └──────────────────────────────────────────┘
                      │
                      ▼
            ┌──────────────────┐
            │ His_Acknow Table │
            └──────────────────┘
```

Figure 50 Information Class Diagram

```
  ┌─────────┐            ┌──────────────────────────────────────────┐            ┌───────────┐
  │ JFrame  │──────────▶ │                ShowTable                 │ ◀──────────│  MainTask │
  └─────────┘            ├──────────────────────────────────────────┤            └───────────┘
                         │ table:JTable                             │              (uses)
  (specialization)       │                                          │
                         ├──────────────────────────────────────────┤
                         │ ShowTable();                             │
                         │ ShowTable(int,int);                      │
                         │ displayResultSet(ResultSet);             │
                         │ schoolName(ResultSet):String[];           │
                         │ getNextRow(ResultSet,ResultSetMetaData):Vector; │
                         │ transToResult(int,int):String;           │
                         │                                          │
                         └──────────────────────────────────────────┘
```
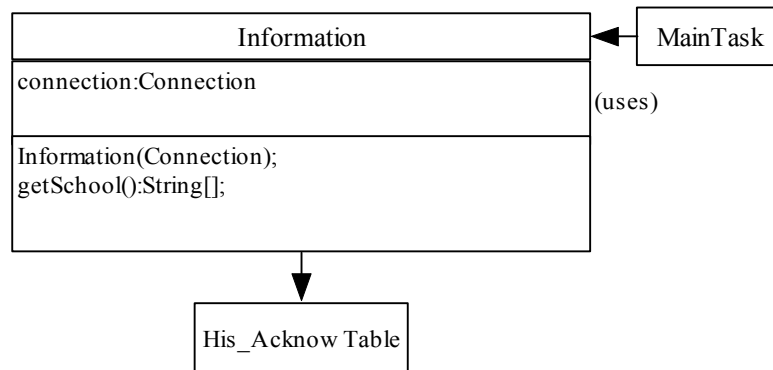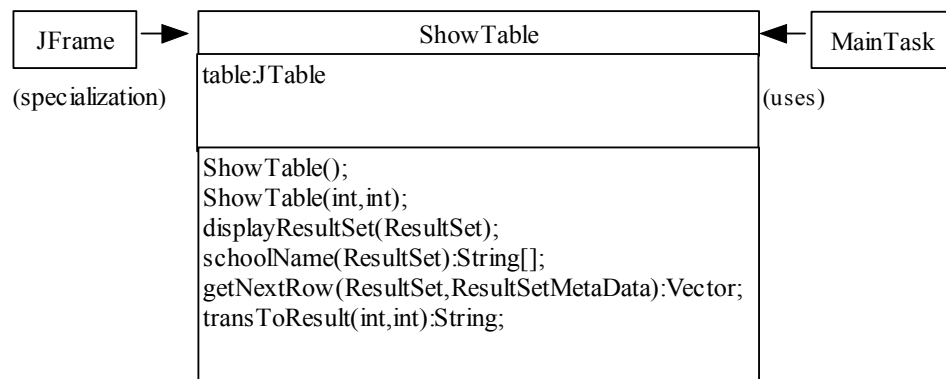
Figure 51 ShowTable Class Diagram

3.3 Interface program and Database Package

3.3.1 Interface package

The HTML files in the Interface package interact with back-end servlet programs. The organizations of these files are presented in Figure 50. The design and implement of these files are similar. So we describe two of them- Login.html and selectQuestions.html in Figure 51 and Figure 52, respectively.
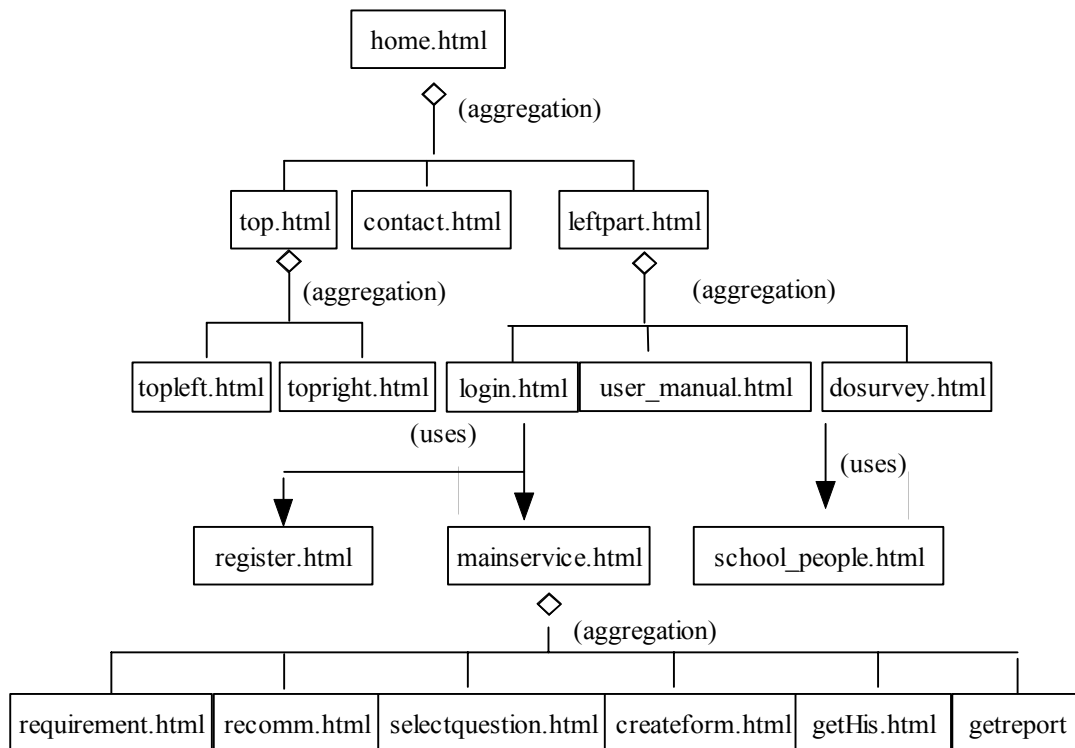
```
                          home.html
                             ◇ (aggregation)
          top.html    contact.html    leftpart.html
             ◇ (aggregation)              ◇ (aggregation)
      topleft.html  topright.html   login.html  user_manual.html  dosurvey.html
                              (uses)                                    (uses)
                    register.html    mainservice.html    school_people.html
                                        ◇ (aggregation)
  requirement.html  recomm.html  selectquestion.html  createform.html  getHis.html  getreport
```

Figure 52 Construct of package of HTML file

```
┌──────────────────────────────────────┐        ┌──────────────────┐
│              login.html              │───────▶│ LoginServlet.java│
├──────────────────────────────────────┤ (uses )└──────────────────┘
│ UerName:String                       │                 │
│ password:Password                    │                 │
│                                      │                 │
│                                      │                 │
├──────────────────────────────────────┤                 │
│ sendform(form);                      │                 │
│ reset(form);                         │                 │
│ focus();                             │                 │
│                                      │                 │
│                                      │                 │
│                                      │                 │
└──────────────────────────────────────┘                 │
                    ◇ (aggregation)                       │
         ┌──────────┴──────────┐                          │
   ┌──────────────┐    ┌──────────────┐                   │
   │ register.html│    │ mainservice  │◀──────────────────┘
   └──────────────┘    └──────────────┘
```
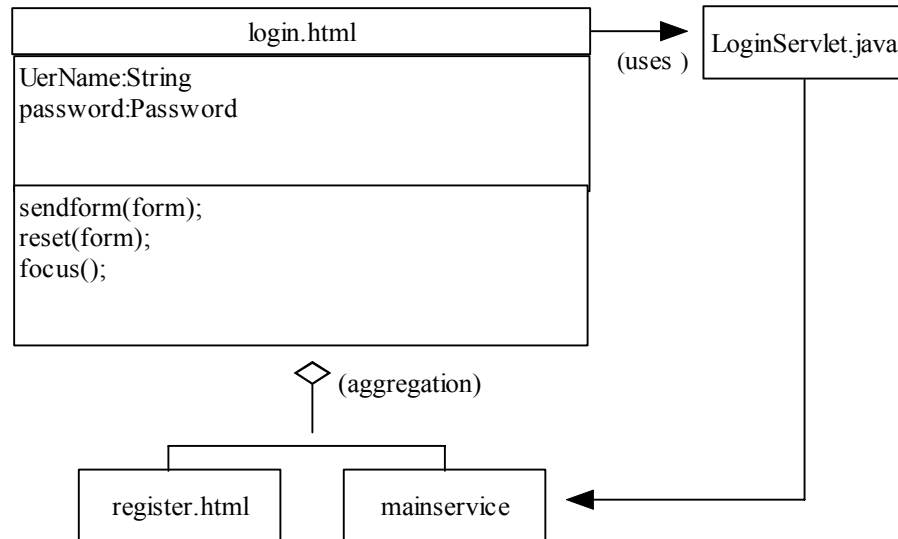
Figure 53 Login.html Class Diagram

```
┌──────────────────────────────────────┐        ┌──────────────────┐
│          selectquestion.html         │───────▶│ SelectQuestionSe │
├──────────────────────────────────────┤ (uses) │ rvlet.java       │
│ schoolName:String                    │        └──────────────────┘
│ peopleCategory:String                │
│ survey0,survey1,survey2,survey3,survey4, │
│ survey5,survey6,survey7,survey8,survey9, │
│ survey10,survey11,survey12:String[]  │
│ surveyindex:int                      │
├──────────────────────────────────────┤
│ sendform(form);                      │
│ makearray();                         │
│ openWin(url);                        │
│ addStuff(form,String[],int[]);       │
│ remove(form,String,int[]);           │
│ showStuff(String,int[]);             │
│ showAll();                           │
│ showNumber(form,String[],int[]);     │
│                                      │
└──────────────────────────────────────┘
```

Figure 54 Selectquestion.html Class Diagram

3.3.2 Database Package

The description of each of the 9 basic tables in SSIOSS Database is presented in

Tables 1-9.

Table 1. Registration Table Definition

Field Name	Data Type	Description
UserID	Text	User ID (primary key)
Password	Text	Password
SName	Text	School name (primary key)
SAddress	Text	School address (street)
City	Text	City
State	Text	State
Zip	Text	Zip code
PfName	Text	Principal first name
PlName	Text	Principal last name
PPhone	Text	Principal phone number
PEmail	Text	Principal e-mail address
CfName	Text	Contact person first name
ClName	Text	Contact person last name
CPhone	Text	Contact person phone number
CEmail	Text	Contact person e-mail address

Table 2. Requirement Table Definition

Field Name	Data Type	Description
SName	Text	School name (primary key)
Student	Number	Do or not do a student's survey
Parent	Number	Do or not do a parent's survey
Teacher	Number	Do or not do a teacher's survey
Staff	Number	Do or not do a staff's survey
Year	Number	Which year Did your school survey at BESAR
BGStudent	Number	Do you need to compare the answers of boys and girls?
MFTeacher	Number	Do you need to compare the answers of male and female teacher?
FIParent	Number	Do you need to compare the answers of people who have different family income?
GStudent	Number	Do you need to compare the answers of student who are in different grades?
LTeacher	Number	Do you need to compare the answers of elementary, secondary, and high school teacher?
All	Number	Do you need to do all above cooperation?
Years	Number	Do you need to compare answers of different year survey?
CurrentTime	Text	Current time you have give up the requirement

Table 3. Form Table Definition

Field Name	Data Type	Description
SName	Text	School name (primary key)
People	Text	Survey form that are for these group of people
Location	Text	The location where survey form are stored.

Table 4. ItemBank1 Table Definition

Field Name	Data Type	Description
SName	Text	School name
People	Text	People that these set of questions serve to.
IM	Text	Section name: Instructional Management
IM-1	Text	IM question 1
……	Text	Questions are skipped from question 2 to 17 at here
IM-18	Text	IM question 18
TL	Text	Section name: Teaching and Learning
TL-1	Text	TL question 1
……	Text	Questions are skipped from question 2 to 13 at here
TL-14	Text	TL question 14
AP	Text	Section name: Academic Profile
AP-1	Text	AP question 1
……	Text	Questions are skipped from question 2 to 55 at here
AP-56	Text	AP question 56
SB	Text	Section name: Student Behavior/Values
SB-1	Text	SB question 1
……	Text	Questions are skipped from question 2 to 17 at here
SB-18	Text	SB question 18

Table 5. ItemBank2 Table Definition

Field Name	Data Type	Description
SName	Text	School name (primary key)
People	Text	People that these set of questions serve to. (primary key)
A	Text	Section name: Administration
A-1	Text	A question 1
……	Text	Questions are skipped from question 2 to 11 at here
A-12	Text	A question 12
G	Text	Section name: Guidance
G-1	Text	G question 1
……	Text	Questions are skipped from question 2 to 5 at here
G-6	Text	G question 18
TSR	Text	Section name: Teacher-Student Relationships
TSR-1	Text	TSR question 1
……	Text	Questions are skipped from question 2 to 8 at here
TSR-9	Text	SB question 9
SA	Text	Section name: Student Activities
SA-1	Text	SA question 1
……	Text	Questions are skipped from question 2 to 6 at here
SA-7	Text	SA question
SSM	Text	Section name: Support Services/Maintenance

SSM-1	Text	SSM question 1
......	Text	Questions are skipped from question 2 to 12 at here
SSM-13	Text	SSM question
PCSR	Text	Section name: Parent-Community-School Relationship
PCSR-1	Text	PCSR question 1
......	Text	Questions are skipped from question 2 to 11 at here
PCSR-12	Text	PCSR question
OR	Text	Section name: Overall Rating
OR-1	Text	OR question 1
OR-2	Text	OR question 2
OR-3	Text	OR question 3

Table 6. History Table Definition

Field Name	Data Type	Description
SName	Text	School name (primary key)
Time	Text	When school user used system? (primary key)
Purposes	Text	What school user did?

Table 7. Report Table Definition

Field Name	Data Type	Description
SName	Text	School name (primary key)
Year	Text	Year of this report for

Location	Text	Location of this report are stored

Table 8. History Acknowledgement Table Definition

Field Name	Data Type	Description
SName	Text	School name (primary key)
Regis	Number	Do school have registers or not?
Require	Number	Do school give up requirement or not?
SQ1	Number	Survey Questions have selected for student or not?
SQ2	Number	Survey Questions have selected for teacher or not?
SQ3	Number	Survey Questions have selected for parent or not?
SQ4	Number	Survey Questions have selected for staff or not?
SF1	Number	Survey Form has created for student or not?
SF2	Number	Survey Form has created for teacher or not?
SF3	Number	Survey Form has created for parent or not?
SF4	Number	Survey Form has created for staff or not?
ST1	Number	Survey Table has created for student or not?
ST2	Number	Survey Table has created for teacher or not?
ST3	Number	Survey Table has created for parent or not?
ST4	Number	Survey Table has created for staff or not?
Report	Number	Report has finished for this school or not?

Table 9. Recommendation Table Definition

Field Name	Data Type	Description
Index	Number	Index of plan (primary key)
Require	Text	Goal of required recommendation by school user
Condition	Number	The requirement (condition) to achieve the goal
Plan	Number	Plan number
Answer	Text	Plan statement

CHAPTER IV: Testing by Example

Examples of the basic services offered by the SSIOSS system to school user, end-user, and administrator are included here. Unit testing, black box testing, integrated testing and system testing were conducted for this project. The black box testing and example of SSIOSS together was combined as follows.

4.1 Home WebPages

The SSIOSS system started with home web page. When the address-http://134.129.189.111:8080//.. webpages/newversionproject/home.html in address bar of Internet Explorer or Netscape Navigator was typed in, the SSIOSS system home page appeared as showing Figure 55.
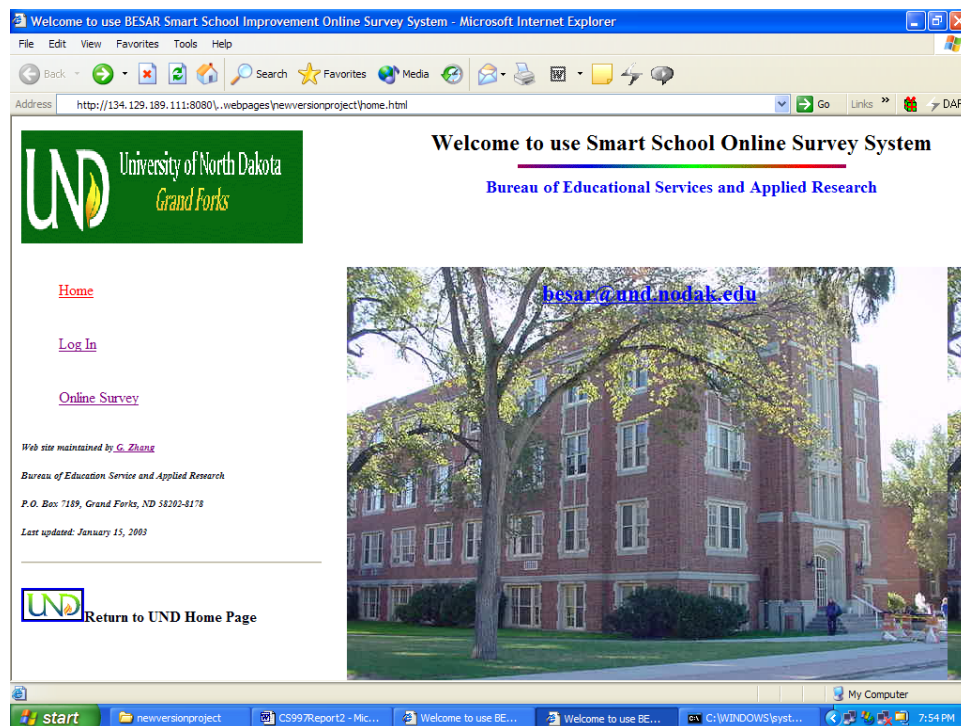
Figure 55 SSIOSS system home page

This was tested in both Internet Explorer and Netscape Navigator under Windows 98/2000. In this page, click home to view user manual. Press Log In to login without typing user name and password for school user. Press Online Survey to start online survey for end user. Send E-mail to BESAR by checking besar@und.nodak.edu.

4.2 School User Operation Example

4.2.1 Login and registration

After Log In was pressed, the Login Form appeared in Figure 56. The registered school users may login with their user ID and password. Login by correct and incorrect user name and password were tested, and these school user names have already stored in registration table in SSIOSS database.

Figure 56 Login interface

If you were a new school user, please press Register on Login Form. The Registration Form appeared in Figure 57. You were required to enter school name, user name, and password. School name, user name and password were used to identify the

user, which should not be the same for different schools and school users. Otherwise error message could be sent out by the system.

Figure 57 Registration Form

4.2.2 Main service interface

After login successfully, the main service interface for school user was displayed. The listed main services provided for school user was presented in figure 58.The main service page will keep the userID and password as session identification when data input was in operation. You must logout after data input was finished or you wanted to quit SSIOSS, otherwise strangers would be able to sneak into the account.
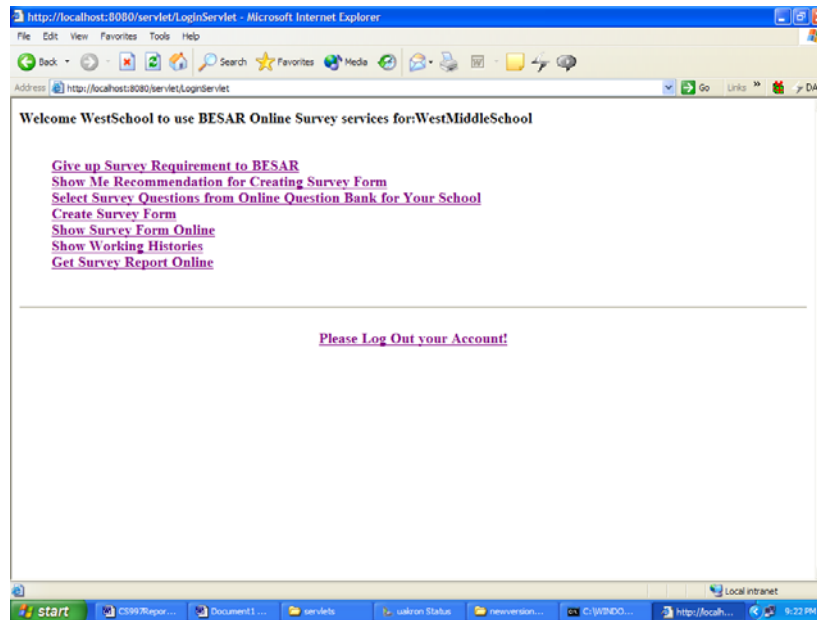
Figure 58 Main service interface for school user

4.2.3 Withdraw requirement for school improvement survey

Press <u>Give Survey Requirement to BESAR</u> on the main service interface page.

The form to select the requirement for a particular survey was presented in Figure 59.

Figure 59 Form of requirement for survey

The first requirement is multiple choices for group of people included in the survey. Second requirement is the year survey was conducted at BESAR. If the school was a new customer, current year was designed to select. Third requirement is multiple choices for comparison of items responded by the group difference in the data analysis process. As was expected, the choice would be dependent on the first requirement and would affect the survey process.

4.2.4   Select survey items to a survey form

After requirements were sent out, survey items were selected. BESAR stored a survey item bank for school improvement evaluation purpose. They are kept in SSIOSS database. So when the link Select Survey Questions from Online Question Bank for Your School was pressed in main service page, the selected item form appears as showing in Figure 60.

Figure 60 Form of Selected Items of SSIOSS

Items were selected in the same item section. For example, a question was selected elect in your survey form, then press ADD button that is above the select group to add this question to the survey form. Repeat this step for each section of questions, and finally when you submit the form to BESAR, don't forget typing in school name and selecting category of people.

4.2.5   Create survey form, survey table, and back-end servlet program

After survey items were selected, press <u>Create Survey Form</u> in the main service page, the created form page appeared as shown in Figure 61. Select category of people and submit your form. SSIOSS system will automatically create survey form, and generate data table, back-end servlet program.
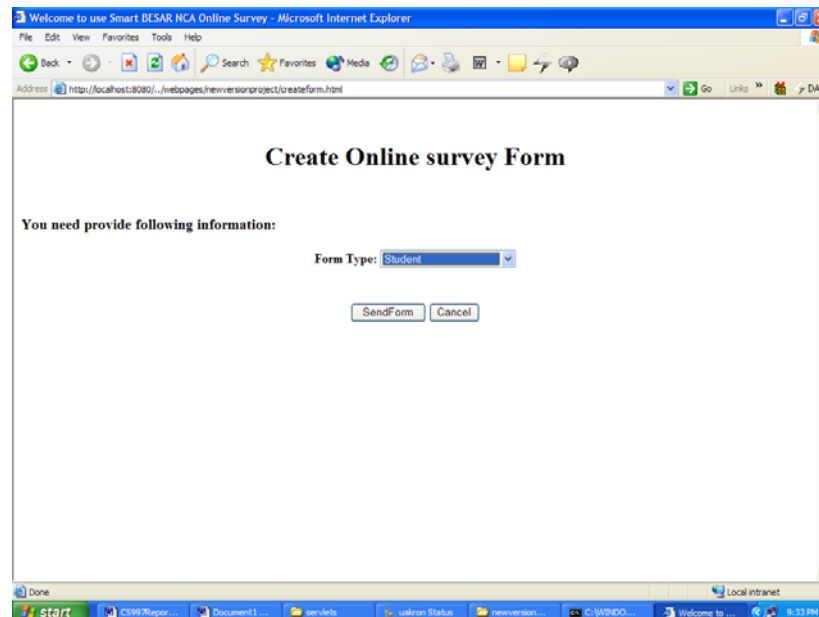
Figure 61 Create survey form interface

4.2.6 Get recommendation from SSIOSS

Section 4.2.1 to 4.2.5 are necessary steps to follow. Otherwise, some-error may occur when the school user was working on the interface, such as if you forgot what jobs you did from previous login or you did not know what you should do in the next step.

The recommendation was design for the system with such situations. Press <u>Get</u>

<u>Recommendation</u> in main service interface, the recommendation page appeared in Figure

60. There were three choices in the recommendation interface. They were: 1) what next

step I need to do? 2) How much I need to pay? 3) How long can I get the report? Select

one of them to try for an answer from the system.
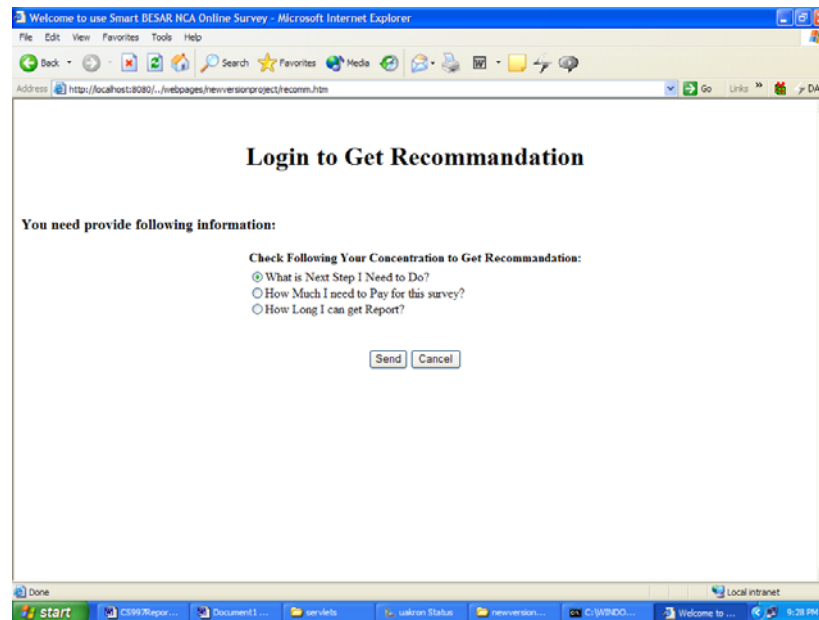
Figure 62 Recommendation interface

4.2.7 Working history identification

Press <u>Show Working Histories</u> in main service interface. West Middle School

survey school user working histories was demonstrated in Figure 63 as an example.

Figure 63 Answers of Show working histories

4.2.8 Get survey report from SSIOSS

Press Get Survey Report Online in main service interface. Your school survey report appear, right click mouse and select save target as to download the report to your local computer.

4.3 End-user operations

End-users are those who respond to the survey items online for the school. End – user may be a student, a teacher, a parent, or a staff in the school. When an end-user wants to answer questions, just opens the SSIOSS home page and press Online Survey. The interface will appear, Type in school name and select category of people, and then submit the form. The question form of West Middle School was demonstrated in Figure 64 as an example. Sometimes, a long question form was included with items that both of an individual and the school were interested in the answer. Follow the form to answer each of the questions and submit the form when they were finished it.

53

Figure 64 Example of Student Question form for West Middle School

4.4   Administrator operations

4.4.1   Main service of administrator

The administrator is a data analyst working for the BESAR or a SSIOSS system, this person's job is on the server computer. To start administrator operations, the first step is to get to the administrator under C:/jswdk directory. Then double click StartAdministrator.bat file; the main service interface for administrator was presented in Figure 65. Right part of figure 65 is where you can find StartAdministrator file. Left –top part is main service interface for administraotor. Left- bottom part is DOS Command Prompt after StartAdministrator was double clicked. There are five services in Administrator main service interface. Two of them are the most useful: List School and List Process. Press button List School will list all school names currently using SSIOSS. Press button List Process will list the status of all the schools in the process of the school

improvement survey. The results were presented in Figure 66. Top part of figure 66 is a

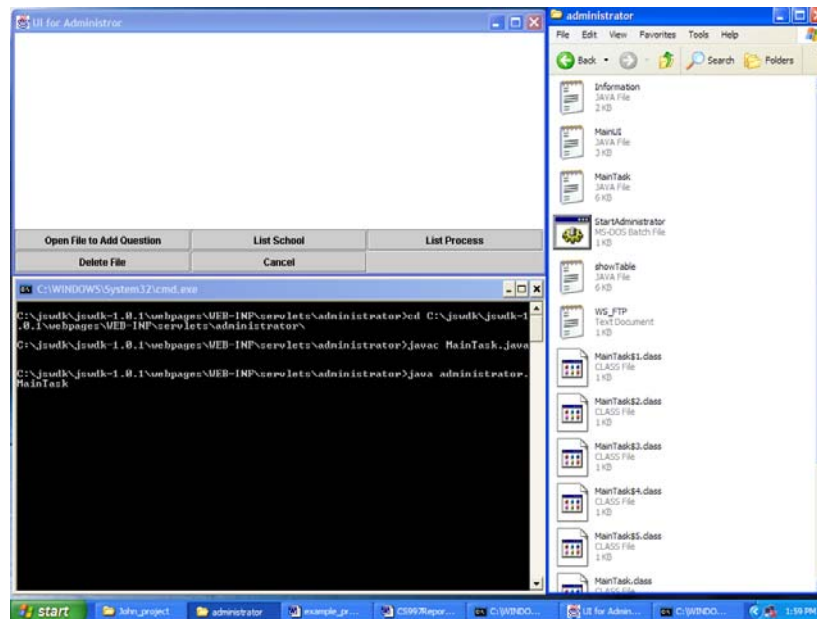result of List School. Bottom part of figure 66 is a result of List Process.

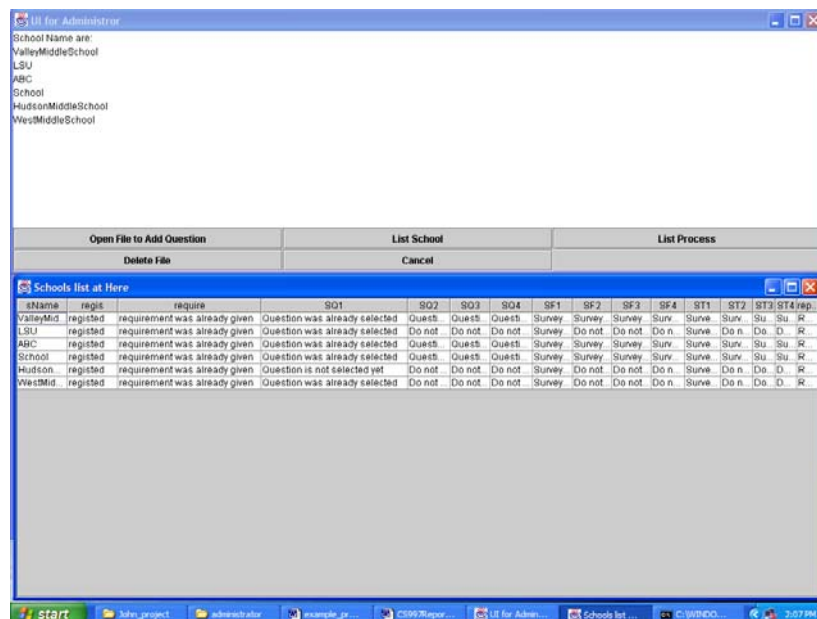Figure 65 How to get administrator main service interface

Figure 66 Results of List School and List Process

55

4.4.2 Compile automatically generated back-end servlet file

Under DOS Command Prompt, type javac C:/jswdk/jswdk-1.0.1/webpages/WEB-INF/servlets/ and full file name for compiling.

4.4.3   Add school improvement survey report to database

After step of survey data collection, data are analyzed by using SPSS and survey report is generated. Final basic task is adding survey report to database by administrator. Open table report in SSIOSS database. Type school name, year of survey report for, and the name of survey report including full path of location to the table and save it.

CHAPTER V: Conclusion and Future Work

The SSIOSS project has created the means for online school improvement survey to support school user operations. These operations enable SSIOSS to work efficiently with schools, enabling them to professionally generate their own survey form with the direct guidance of BESAR's expertise on school improvement survey, but without the need for face to face consulting. Also our use of the Belief Desire Intention (BDI) model to implement SSIOSS made the system design clearer.

Automating the generation of the survey form, survey table, and back-end program also is time and cost efficient for the Survey Company. This is another characteristic property of SSIOSS.

In this report, we have described in detail the project requirement specification based on the BESAR's measurement and evaluation expertise on school improvement survey. Also we have used Class Diagram of UML model to describe project design process that match requirement specification to program code. This is very helpful information for programmers who to implement complementary projects.

At this time, the data analysis steps have not been automated. It is recommended that SAS data analysis procedures should be integrated into the SSIOSS system. This implementation would allow the system automatically to analyze data stored in server database. Right now, the major software for data analysis for the BESAR is SPSS. This has some limitations for implementing such a project. This is the one of our recommendations for future improvements to SSIOSS.

REFERENCES

[1] Bratman, Michael E., Intention, Plans, and Practical Reason, Harvard Univ. Press, 1987 (also by CSLI Publication, 1999).

[2] Deitel, H. M., Deitel, P. J., Java How to Program, third edition, Prentice Hall, 1999.

[3] Elmasri, Ramez, Navathe, Shamkant B., Fundamentals of Database Systems, third edition, Addison-Wesley, 2000.

[4] Ghezzi,Carlo, Jazayeri, Mehdi, Mandrioli, Dino, Fundamentals of Software Engineering, second edition, Prentice Hall, 2002.

[5] NCA Standards and Criteria for Elementary, Middle Level, Secondary and Unit Schools, Tempe, AZ North Central Association Commission on Schools, 2000.

[6] Pressman, Roger S., Software Engineering, fifth edition, McGraw-Hill Higher Education, 2001.

APPENDICES

APPENDIX 1: Start School Improvement Online Survey System User Manual

APPENDIX 2: Program Source Code

APPENDIX 1: Smart School Improvement Online Survey System (SSIOSS) User Manual (Readme.doc)

1. Introduction

Smart School Improvement Online Survey System (SSIOSS) is an Internet-based application project used for school improvement survey written by Java programming language. SSIOSS provides a set of basic services. These include four main parts. They are school user making survey form, school user getting recommendation, administrator checking school user working condition, and survey user answer survey online. Figure 67 presents the structure of SSIOSS packet and contents. Figure 68 presents the content of Code package in SSIOSS. Figure 69 presents the content of interface and database package in SSIOSS.

1. School user makes survey form.

This is a set of operation in the packet of form and servlets. It includes school user registration, login, give up requirement, select survey question, create survey form, create back–end program, create survey table (database), and compile program.

2. School user gets recommendation.

These of operations are in packet of recomm and servlets. I use a concept of Belief, Desire, and Intention (BDI) as a model to implement this part of project. The major operation are what next step the school user getting to know and need to do, how much the school user need to pay and how long the school user can getting report. Also school user can get the history that he /she did for the school in the processes of creating survey form.

60

3. Administrator checks school user's working condition and job.

These are including check school user list, school user working history, adding and delete file. These classes are in packet of administrator.

4. Users do the survey online.

After school user creates survey form, back-end program, and tables of database and compile it. The end user can use these programs to do the online survey. The end user may be student, teacher, parent or school staff. They can use the program from their own computer or public computer from anywhere just need getting to the web. The survey forms and tables are in packet of newversionproject. The back-end programs are in packet of servlets. These programs are automatically generated in provirus process.
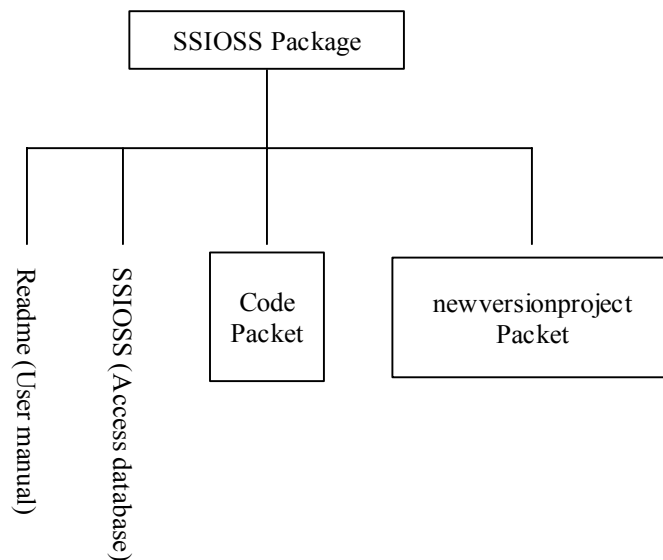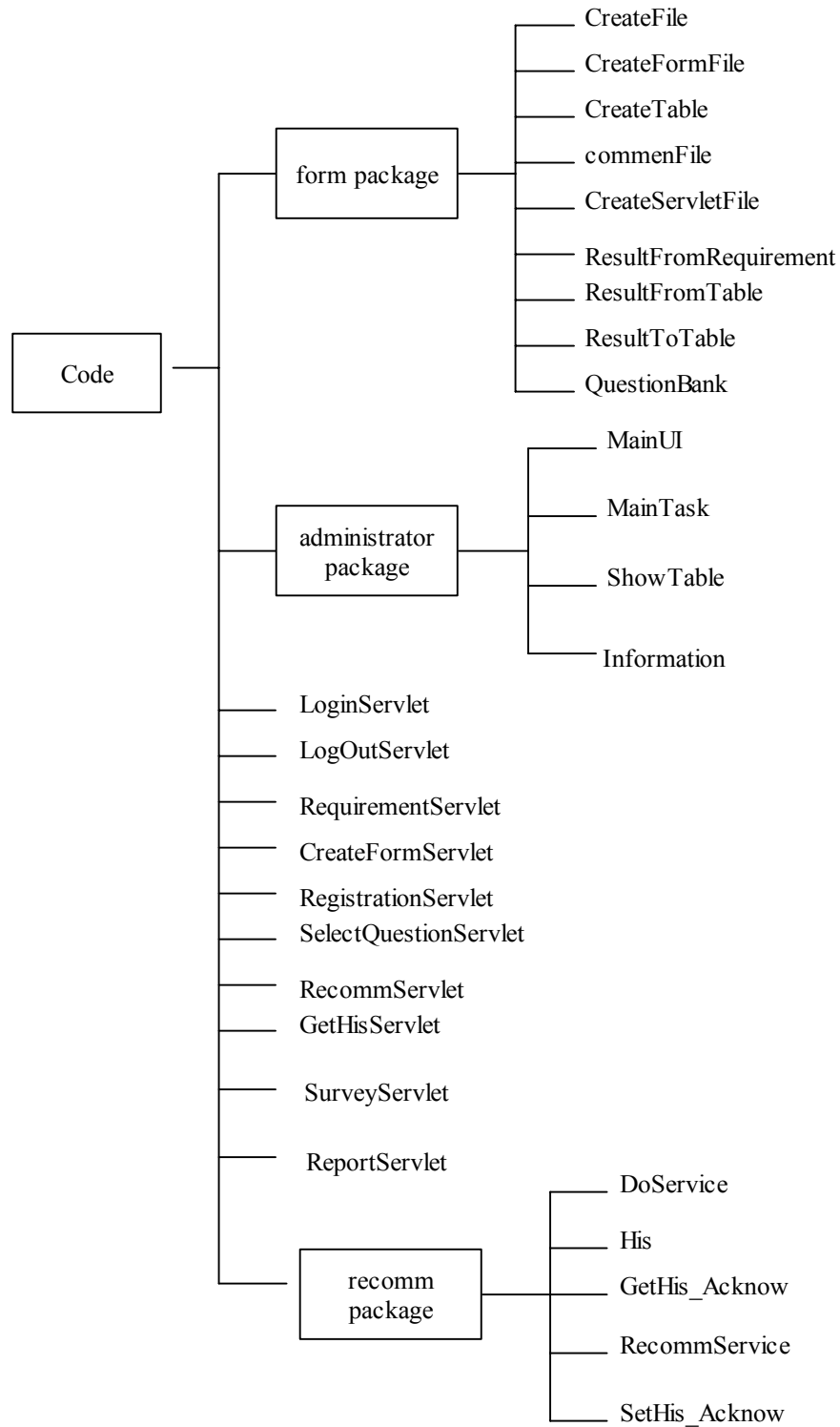

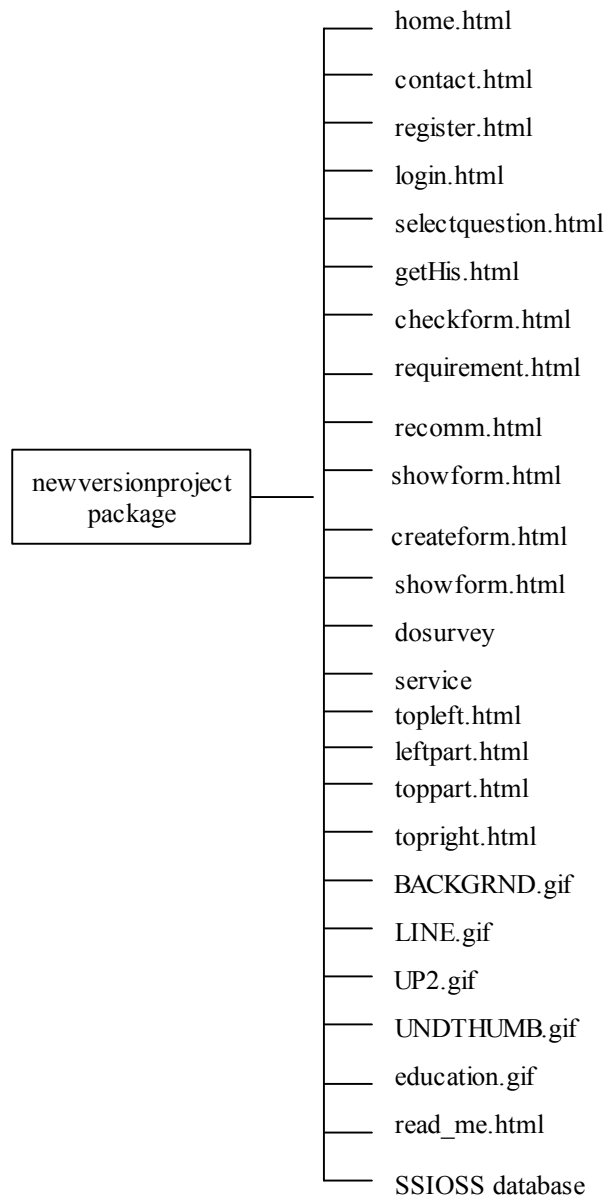Figure 67 SSIOSS package structure

Figure 68 Code package in SSIOSS

```
                                  home.html

                                  contact.html

                                  register.html

                                  login.html

                                  selectquestion.html

                                  getHis.html

                                  checkform.html

                                  requirement.html

                                  recomm.html

  ┌─────────────────┐             showform.html
  │ newversionproject│──────┐
  │     package      │      │     createform.html
  └─────────────────┘
                                  showform.html

                                  dosurvey

                                  service
                                  topleft.html
                                  leftpart.html
                                  toppart.html

                                  topright.html

                                  BACKGRND.gif

                                  LINE.gif

                                  UP2.gif

                                  UNDTHUMB.gif

                                  education.gif

                                  read_me.html

                                  SSIOSS database
```

Figure 69 Contents of interface and database package

2. Requirements

- **Operating System**

  This release has been tested on the following platforms: Windows 98 and 2000.

- **JDK**

  Install JDK1.1.8 or up version. I used jdk 1.3.1 and you can download it from:

  http://java.sun.com/products/jdk/1.3/install-windows.html

- **JSWDK1.0.1**

  Download and install JSWDK1.0.1 at http://java.sun.com/

- **Environment variables** [2]

  1. Add the full path of the JDK bin directory (where the java command is located) to the PATH variable. To do so, click Start-> Run and enter Sysedit. Open and edit the AUTOEXEC.BAT file and add or change the PATH statement.

  2. Add the tools.jar (from the JDK lib directory) file to your CLASSPATH variable. Save AUTOEXEC.BAT and close it. Restart your computer.

  3. Copy servlet.jar (from JSWDK1.0.1 lib directory) file to your JDK extensions directory (the directory c:\jdk1.2.2\jre\lib\ext on windows).

- **Microsoft Access**

  To execute this application, you need to have a database system. I used Microsoft Access as the database for this project. ODBC for MS Access database is provided with windows system. You need to set up the SSIOSS database file as described in the following section when you install the system.

3. Install SSIOSS

- **Register SSIOSS database**

SSIOSS database named SSIOSS.mdb in packet of newversionproject and contains basic nine tables originally. SSIOSS will be automatically increasing when school user uses this project to do their own job. From outer side to connect to the database, an ODBC Data Source must be registered with the system through the ODBC Data Sources option in the Windows Control Panel. The following step you need to follow:

1. Double-click ODBC Data Source option to display the ODBC Data Source Administrator dialog.

2. Select the User DSN tab if it is not selected, then click ADD to display the Create New Data Source dialog.

3. Select Microsoft Access Driver and click Finish. Then the ODBC Microsoft Access 97 Setup dialog appears.

4. Click on Select button to display the Select Database dialog. Use this dialog to locate and select the SSIOSS.mdb file. Click OK to return to the ODBC Microsoft Access 97 Setup dialog.

5. Enter the name SSIOSS in the Data Source Name text field. Enter a description as an optional.

6. Click OK to dismiss ODBC Microsoft Access 97 Setup dialog.

7. Notice that the ODBC Data Source Administrator dialog now contains the data source SSIOSS. Click OK to dismiss the dialog.

- **Copy HTML files**

  Copy packet of newversionproject to the directory jswdk-1.0.1/webpages/

- **Install servlet classes**

  Copy all contents in package of SSIOSScode directory to the directory jswdk-1.0.1/webpages/WEB-INF/servlets.

4. Operation procedure for administrator

- **Start server**

  To start the server, open the directory with Windows Explorer and double-click on STARTSERVER.BAT in JSWDK1.0.1 directory

- **Stop server**

  To stop the server, open Windows Explorer and double-click on STOPSERVER.BAT in JSWDK1.0.1 directory.

- **Use administrator MainTask**

  To start using administrator operation, open the directory with Windows explorer and double click on StartAdministrator.bat in administrator directory. Then select task that you want to do from the user interface. The program will follow your task to do delete file, show school name, show school working history, etc.

- **Compile back-end file that school user automatically had generated**

  To compile back-end file that school user automatically had generated, open the directory with Windows explorer and double click on compile_file.bat in administrator directory. Then using command javac filename.java to compile the file that want to compile.

5. Operation procedure for school user

To start the user-interface, open your web browser. Either Internet Explorer or Netscape Navigator will be fine. Use your browser to access the following address: http:// 134.129.189.111:8080 /../ webpages / home.html. Then the main user interface will be appearing. You can see the list function of operation displayed in the left part, and user manual in the right part of this main page. Then double click <u>Log In</u> link to start your working process by following the below steps:

1. Registers for an account. Double click <u>Register</u> link and give up information about your school and yourself to the system. If you already have an account, please skip this step.

2. Login to the system. Type user name and password that you have registered at system and submit it.

3. After successful login, the school user main service interface appears. Please select the task that you want to do for your school. If you do not know how to work with this system or forgot which step you are in. Please check the working history by double click <u>Show Working Histories</u> to get the working histories that you have down. To get the recommendation by double click <u>Show Me Recommendation for Creating Survey Form</u> to know what next step you need to do.

4. Getting recommendation. After recommendation page appears, you can select the task that you want system answer it to you by select it and press send button.

5. Give up your requirement about your school improvement survey to the system. It will be used by system as set of constraints for your survey during you creates survey form.

6. Select survey questions for particular category by double clicking <u>Select Survey Questions from Online Question Bank for Your School</u>. After survey question form appears, you will be required to give school name and select category of people that survey service for. So you need give up set of survey questions for each category of people that you desired to survey.

7. Create survey form. After you make sure the survey questions are properly as your school survey wish to have. You can double click link <u>Create Survey Form</u> and create the survey form. Also it will automatically create table in SSIOSS database and back-end program.

8. After process of survey people answer survey question have finished. You can get survey report from web by login and double clicking link <u>Get Survey Report</u>. Congregations! You have down your job for school improvement online survey.

6. Operation procedure for end user

Do online survey. After school user created survey form, school administrator can let the people that you wish to survey to answer the questions that your school user have already selected. By doing this, let people get to the home page of SSIOSS, double click link <u>Online Survey</u> in left part and give up school name and select category of people when <u>Login Form for Online Survey</u> page appears. Then people will get the survey form, answer each of questions people wish to answer, and finally submit form to the system.

APPENDIX 2: Source Code

Please ask for attached CD.